AD-751 506

DEVELOPMENT OF WEAPON DELIVERY MODELS AND ANALYSIS PROGRAMS. VOLUME II. DOCUMENTATION OF THE ARMAMENT DELIVERY ANALYSIS PROGRAMMING SYSTEM (ADAPS)

A. Ferit Konar, et al

Honeywell, Incorporated
Minneapolis, Minnesota

April 1972

AFFDL-TR-71-123
Volume II

DEVELOPMENT OF WEAPON DELIVERY MODELS AND ANALYSIS PROGRAMS

Volume II. Documentation of the Armament Delivery Analysis Programming System (ADAPS)

A. FERIT KONAR
MICHAEL D. WARD

HONEYWELL INC.

TECHNICAL REPORT AFFDL-TR-71-123, VOLUME II

APRIL 1972

AIR FORCE FLIGHT DYNAMICS LABORATORY
AIR FORCE SYSTEMS COMMAND
WRIGHT-PATTERSON AIR FORCE BASE, OHIO

315

NOTICE

When Government drawings, specifications, or other data are used for any purpose other than in connection with a definitely related Government procurement operation, the United States Government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data, is not to be regarded by implication or otherwise as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

Copies of this report should not be returned unless return is required by security considerations, contractual obligations, or notice on a specific document.

AIR FORCE/56780/2 November 1972 — 600

Unclassified

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author) Honeywell Inc. Systems and Research Division, Research Dept. Minneapolis, Minnesota 55413	2a. REPORT SECURITY CLASSIFICATION Unclassified; 2b. GROUP NA

3. REPORT TITLE
DEVELOPMENT OF WEAPON DELIVERY MODELS AND ANALYSIS PROGRAMS
Volume II. Documentation of the Armament Delivery Analysis Programming System (ADAPS)

4. DESCRIPTIVE NOTES (Type of report and inclusive dates)
Final Technical Report 5 October 1970 - 5 October 1971

5. AUTHOR(S) (First name, middle initial, last name)
A. Ferit Konar
Michael D. Ward

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
April 1972	314	2

8a. CONTRACT OR GRANT NO. F33615-71-C-1059	9a. ORIGINATOR'S REPORT NUMBER(S) 12261-FR1, Vol. II
b. PROJECT NO. 8219	
c. Task No. 821904	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) AFFDL-TR-71-123 , Vol. II
d.	

10. DISTRIBUTION STATEMENT
Approved for public release; distribution unlimited.

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY Air Force Flight Dynamics Laboratory Air Force Systems Command Wright-Patterson Air Force Base, Ohio 45433

13. ABSTRACT

The computer programs which implement the mathematical analyses and models developed in Volume I are described. The programs are developed in Fortran IV language. Extensive use of subroutines is made to provide programming flexibility when considering alternate airframe/dynamics/control points/ measurement system combinations and their effect on weapon-delivery performance. A demonstration example is included in Volume III to illustrate how these programs are used and how the important error contributors to weapon-delivery performance are identified.

DD FORM 1 NOV 65 1473 REPLACES DD FORM 1473, 1 JAN 64, WHICH IS OBSOLETE FOR ARMY USE.

Unclassified

Security Classification

Unclassified

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Weapon						
Delivery modeling						
Error analysis						
CEP						
Optimal control						

*U.S.Government Printing Office: 1972 — 759-487/166

Unclassified

Security Classification

DEVELOPMENT OF WEAPON DELIVERY MODELS AND ANALYSIS PROGRAMS

Volume II. Documentation of the Armament Delivery Analysis Programming System (ADAPS)

A. FERIT KONAR
MICHAEL D. WARD

FOREWORD

This document is the second of three volumes of the final report of a study conducted for the United States Air Force under Contract F33615-71-C-1059, "Development of Weapon Delivery Models and Analysis Programs". Approximately one man year of effort was covered by the contract. It was initiated under Project No. 8219, "Stability and Control Investigations", Task No. 821904, "Flight Control System Analysis," and administered by the Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio. Major Harvey M. Paskin (FGC), and Mr. Alonzo J. Connors (FGC) were project engineers.

The technical work reported was conducted by the Research Department of the Systems and Research Division of Honeywell Inc. Dr. A. Ferit Konar was the principal investigator; Mr. M.D. Ward was the programmer analyst. Dr. G.B. Skelton and Dr. E.E. Yore were project managers. Technical consultation was provided by Dr. Gunter Stein and Mr. C.R. Stone of Honeywell Inc.

The reporting period was October 1970 to July 1971. The report was first submitted in September 1971. The contractor's report number is Honeywell Report 12261-FR1.

The investigators in this study would like to thank Major Harvey H. Paskin f·· his enthusiastic support, for his technical leadership, and for his assistance in obtaining bomb data. The investigators would also like to thank Mr. Alonzo J. Connors for providing direction and assistance in testing the analysis programs.

This technical report was reviewed and is approved.

C.B. Westbrook
Chief, Control Criteria Branch
Flight Control Division
Air Force Flight Dynamics Laboratory

TABLE OF CONTENTS

TABLE OF CONTENTS--CONTINUED

TABLE OF CONTENTS--CONCLUDED

LIST OF ILLUSTRATIONS

LIST OF ILLUSTRATIONS--CONTINUED

LIST OF ILLUSTRATIONS--CONTINUED

LIST OF ILLUSTRATIONS--CONTINUED

LIST OF ILLUSTRATIONS--CONCLUDED

LIST OF TABLES

LIST OF TABLES--CONCLUDED

SECTION I
INTRODUCTION

Computer programs for the precision weapon delivery system models and performance optimization algorithms reported in Volume I are documented in this volume.

The computer programs are developed in Fortran IV language. The overall program is called ADAPS -- Armament Delivery Analysis Programming System. The system's greatest asset is its flexibility, obtained by extensive use of subroutines.

Each section is written with the user in mind. Enough detail is given so that basic Fortran knowledge is sufficient to understand the majority of the programs. In the documentation of each program, first input/output information is given. Subsequently flow charts, and source program listings are presented. In addition, a table of symbols is provided for many programs.

Section II provides a brief description of the overall organization of ADAPS. Section III documents ADAP 1 -- Main Program for Nonlinear Simulation and Linearization, and Section IV documents ADAP 2 -- Main Program for discrete optimization of Nonstationary systems. This is followed by the documentation of ADAP 3 -- Main Program for Nonstationary Performance Evaluation of Weapons in Section V. Section VI summarizes the programming and documentation work and lists recommendations for additional areas of programming and extensions to weapon systems study by ADAPS.

The documentation of table-input-lookup-and interpolation processes is given in Appendix I. The documentation of DIAK -- Program for Optimization of Stationary Systems -- is given in Appendix II.

A demonstration example is included in Volume III to illustrate how these programs are used and how the important error contributors to weapon delivery performance are identified.

SECTION II
OVERALL ORGANIZATION OF ADAPS

This section is a brief discussion of the overall structure of the Armament Delivery Analysis Programming System (ADAPS).

The various subroutines implementing the precision weapon delivery models and optimization algorithms provide the capability to analyse weapon delivery as a general linear time-varying, stochastic process or to analyze it as a much simplified process that is stationary during all of its phases. The one extreme offers fidelity to the physical situation, the other offers low computing costs and the possibility of many analysis iterations. By using the program organization shown in Figure 1, both extremes (as well as the many possibilities in between) are readily attainable. In this organization. ADAPS is divided into three groups:

- ADAP 1 -- Simulator and Linearizer
- ADAP 2 -- Optimizer
- ADAP 3 -- Performance Evaluator

The individual subroutines within each group are accessible from a main program with which they share common memory. They communicate with each other within the groups indicated. Optional inputs are provided to cover various analysis objectives.

A typical analysis proceeds as follows:

- Linear Data Generation - This procedure requires the following steps:

  1. Read input data for attack maneuver, read nonlinear aircraft aerodynamics.
  2. Trim aircraft, and fly it to obtain nominal trajectory up to nominal release altitude.
  3. Linearize the aircraft equations of motion numerically at specified time points during flight. Write on tape.
  4. Read input data for nonlinear weapon aerodynamics.
  5. Using the release conditions, generate free-fall trajectory by the nonlinear weapon model.
  6. Linearize the weapon equations of motion numerically at specified time points along the free-fall trajectory, write on tape.

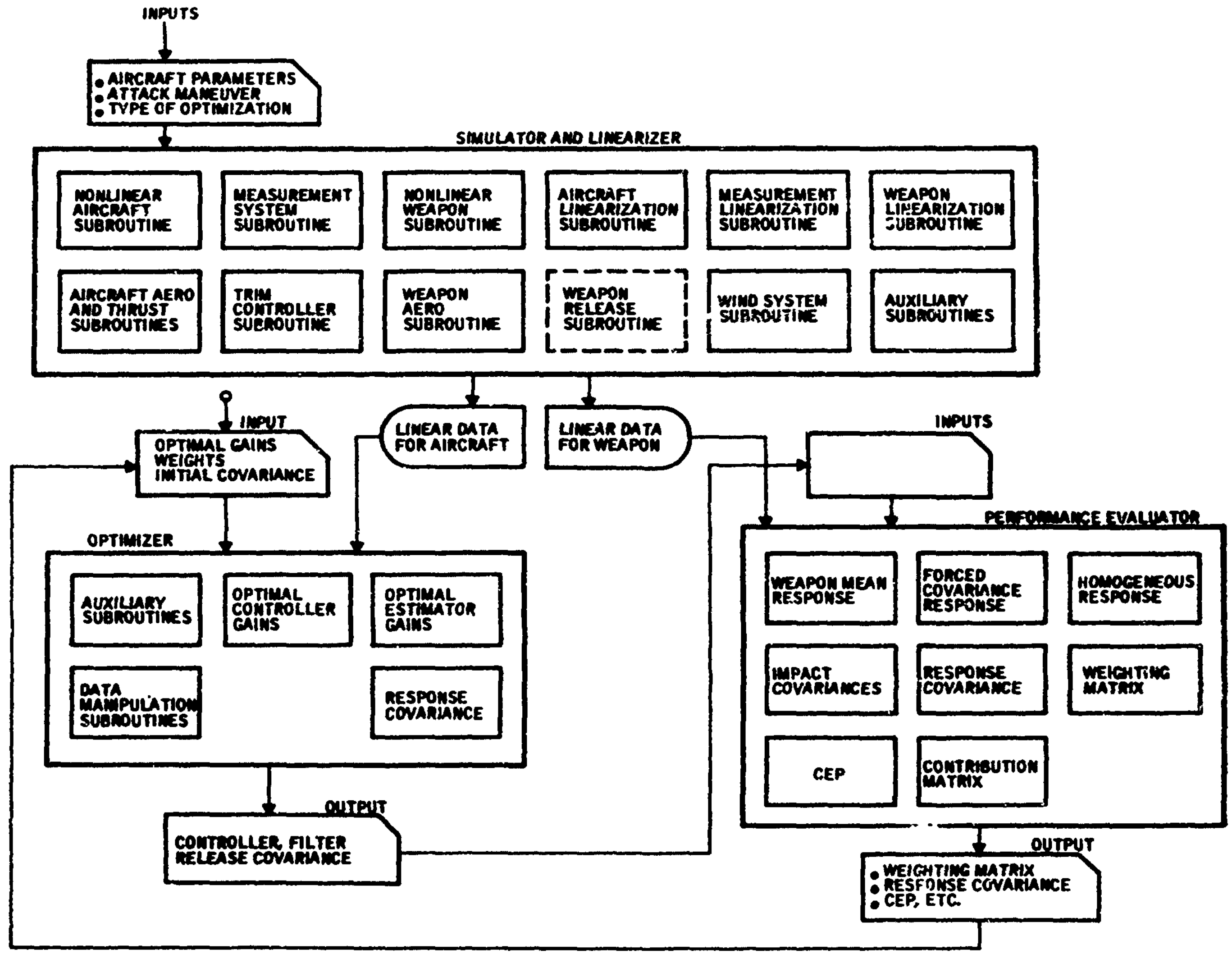


Figure 1. Armament Delivery Analysis Programming System (ADAPS) Overall Organization

Figure 2 is the input/output block diagram of the linear data generation by ADAP 1. Options are also available within the program (aircraft only, weapon only, simulation only, etc.).

- Optimization - This procedure requires the following steps:

  1. Generate the propagation weighting matrix using linear weapon data.
  2. Input control points, measurement points, measurement variances.
  3. Choose the type of optimization, and obtain controller gains, estimator gains, total system covariance at release.

  To obtain the weighting matrix, first the initial covariance of free aircraft is propagated to the release point by ADAP 2. The input/output block diagram for this is shown in Figure 3. Subsequently, the release covariance is propagated to impact by ADAP 3, and the weighting matrix is computed. The input/output block diagram for this is shown in Figure 4.

  These data are used in the second run up of ADAP 2 to produce optimal controller and estimator gains as well as release covariance for the optimal system. Again here various options are available (controller only, estimator only, etc.). Figure 5 is the input/output block diagram of the nonstationary optimization phase.

- Performance Evaluation - This procedure requires the following steps:

  1. Propagate the release covariance to impact using performance evaluator.
  2. Compute impact covariance matrix, CEP performance measure and the variance contribution matrix.

  Figure 6 is the input/output block diagram of the performance evaluation procedure.

The above defines one complete cycle of a typical use of ADAPS. Each part can be used independent of the other for different needs. The main programs are largely at the discretion of the user, to be organized as best suits a particular analysis problem.

For a twentieth-order system, the total computing time per program cycle is approximately 15 minutes using a CDC-6600 processor (Table I), and each main program requires less than 32 K of memory.

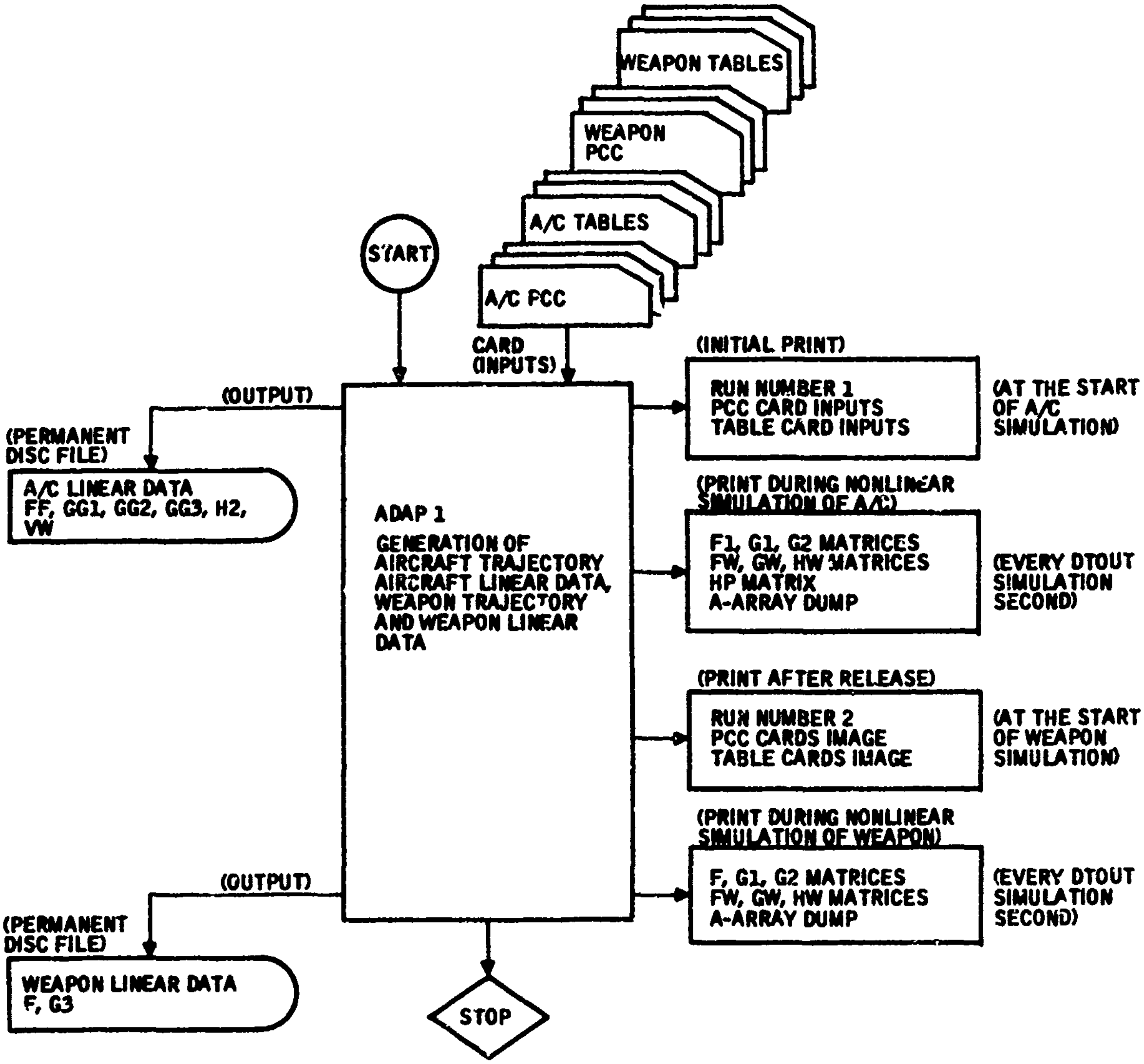


Figure 2. Input-Output Block Diagram of Linear Data Generation Process

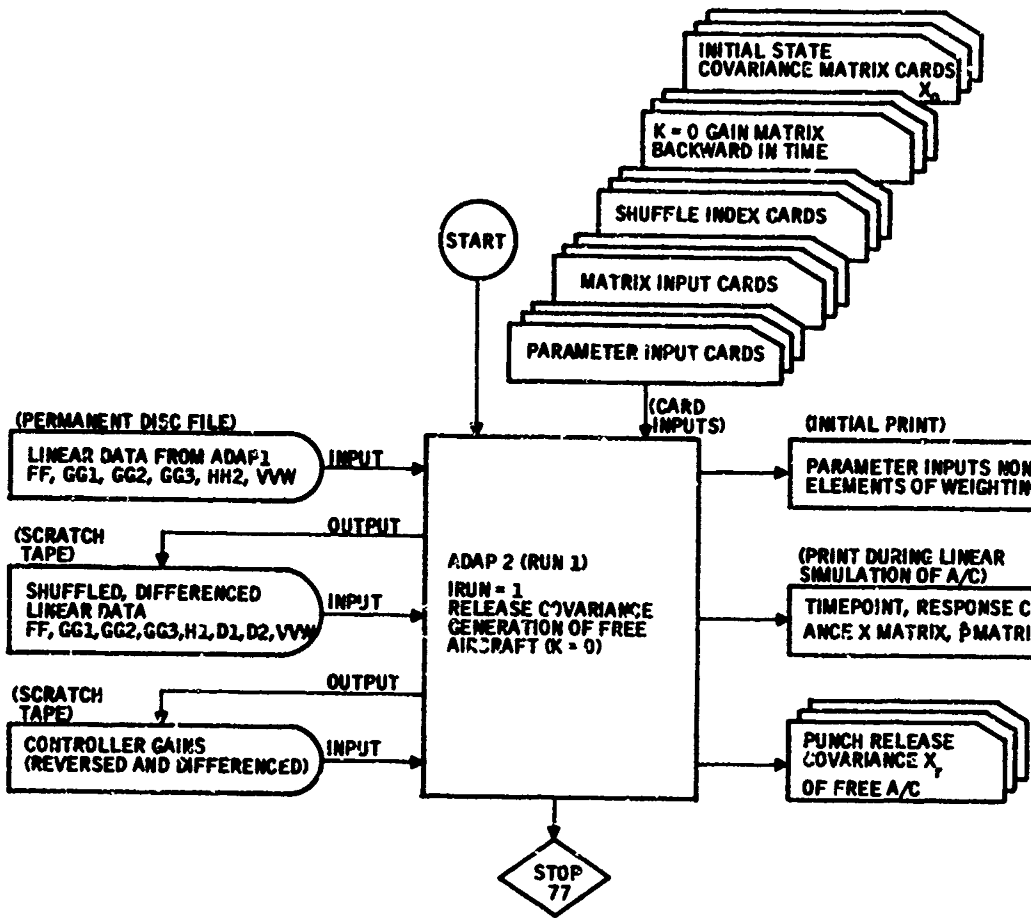


Figure 3. Input-Output Block Diagram of Release Covariance Generation With Optimal Gains

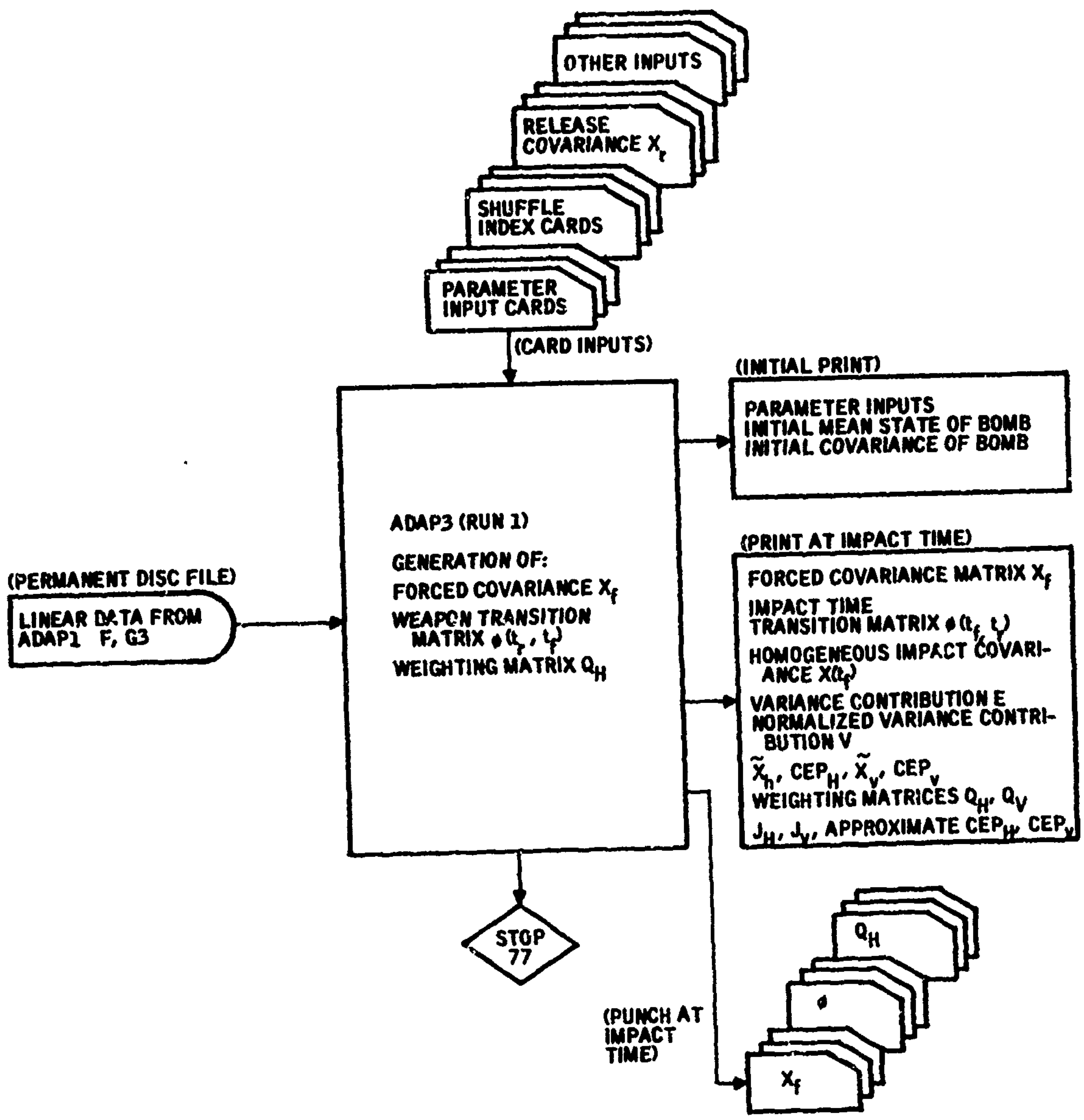


Figure 4. Input-Output Block Diagram of Weighting Matrix Generation

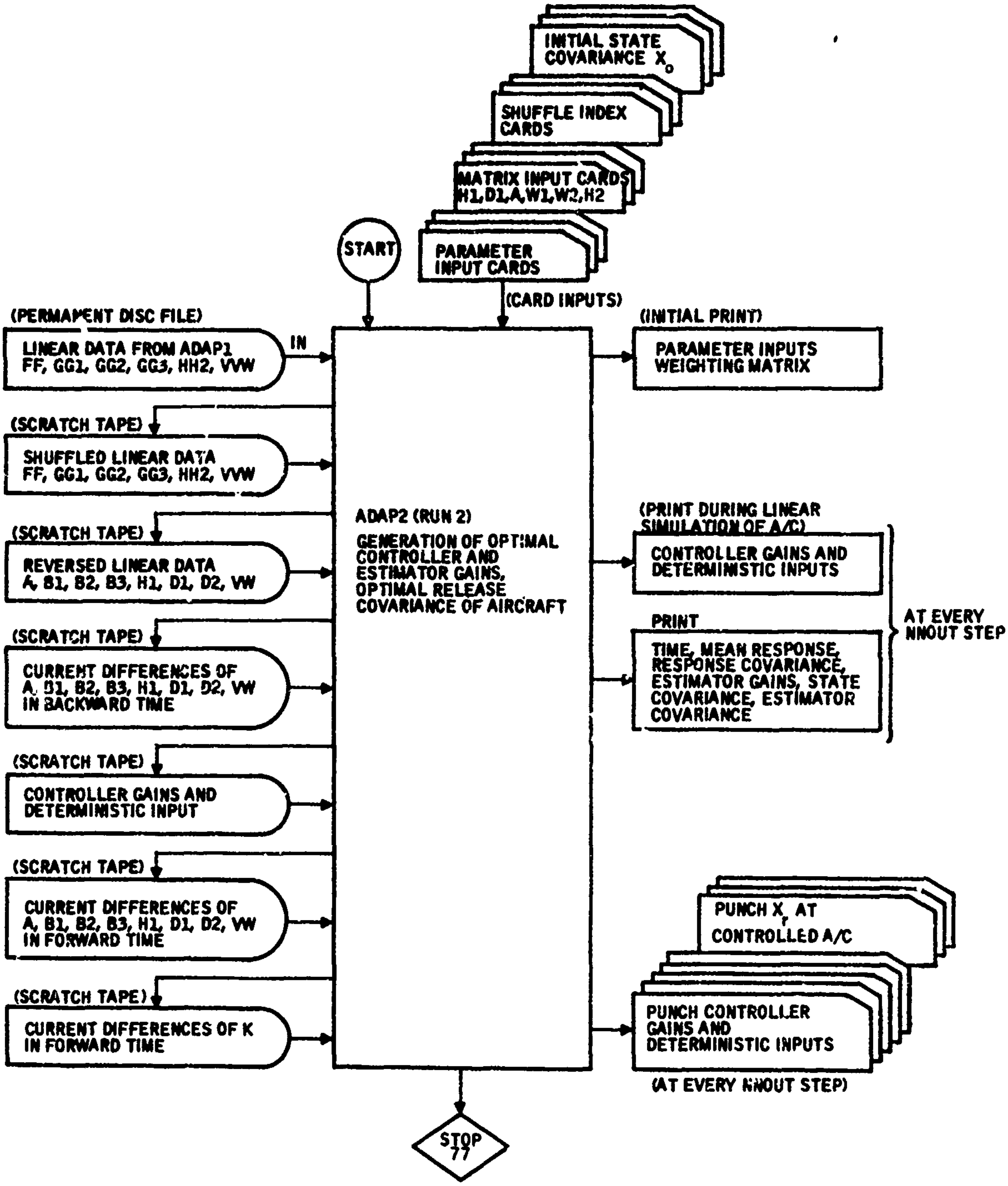


Figure 5. Input-Output Block Diagram of Nonstationary Optimization

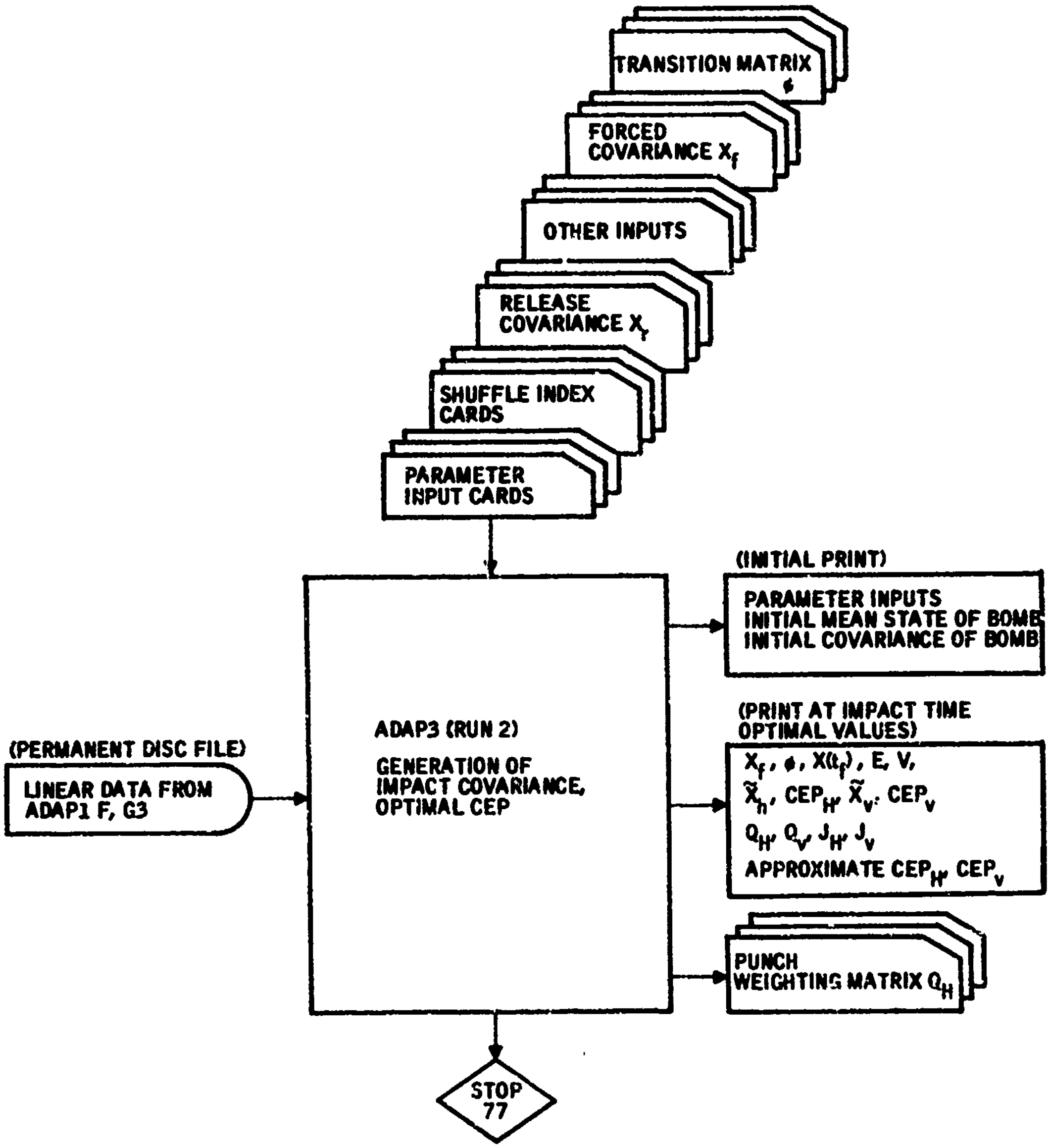


Figure 6. Input-Output Block Diagram of Performance Evaluation

Table I. One Program Cycle of ADAPS

Program	CDC-6600 - Time (sec)			Computational Tasks
	CP	PP	IO	
ADAP 1 (Run 1 and Run 2)	52.250	9.086	2.061	Aircraft dive trajectory, weapon fall trajectory and their linearization
ADAP 2 (Run 1)	124.784	13.835	4.475	Free aircraft covariance at release
ADAP 3 (Run 1)	74.326	3.511	0.722	Weapon covariance at impact and optimal control weighting matrix, corresponding to free aircraft
ADAP 2 (Run 2)	573.020	36.847	14.535	Optimal controller and estimator gains, total covariance at release
ADAP 3 (Run 2)	6.690	2.614	0.636	Weapon covariance at impact and CEP for optimal controller
Total	831.080	65.893	22.429	Approximately 15 minutes/ program cycle

SECTION III

ADAP 1 -- SIMULATOR AND LINEARIZER

ADAP 1 is a six-degree-of-freedom nonlinear simulation and linearization program. It is based on the THRUST program developed by Honeywell [1, 2]. The set of linear data generated by ADAP 1 is used in optimal controller and estimator design (ADAP 2), as well as in weapons delivery performance evaluation (ADAP 3). Figure 7 shows the information flow in ADAP 1 during a simulation.

Briefly, prescribed attack trajectories are flown by simulating an aircraft model using a trim profile input. During the six-degree-of-freedom simulation, the perturbation equations are obtained at specific time intervals by a numerical partial differentiation technique. This process continues until the weapon-release condition is reached. Then free-fall trajectories are generated by simulating a weapon model. During the six-degree-of-freedom weapon simulation, the perturbation equations are obtained at specified time intervals in the same way. This continues until impact occurs.

During the flight from target acquisition to weapon impact, all internal variables of aircraft and weapon and the corresponding linear data are printed out at specified time points. The linear data are also stored in the permanent disk files for subsequent use.

In the balance of this section, input/output information is provided first; then the main program and its subroutines are described.

ADAP 1 INPUT/OUTPUT

A-ARRAY MAP

All information to be transferred between subroutines in ADAP 1, or that is to be available for input and output, is stored in an array. To the computer, an array is a block of storage locations, the first of which is referred to as the base location and identified by some variable name. In the ADAP 1 this name has been designated as A; hence, the term "A-array". All other locations in the array are identified by placing a subscript on A. For example, A(79) represents the 79th location in the A array [2]. There are 1000 entries in the A-array, and they are shared by all ADAP 1 subroutines through the statement

COMMON/ADAP/A(1000)

It is important to record each A-array assignment, and this is done in the A-array map. Table II shows A-array assignments in ADAP 1. The locations with no entry signify their availability for future use.

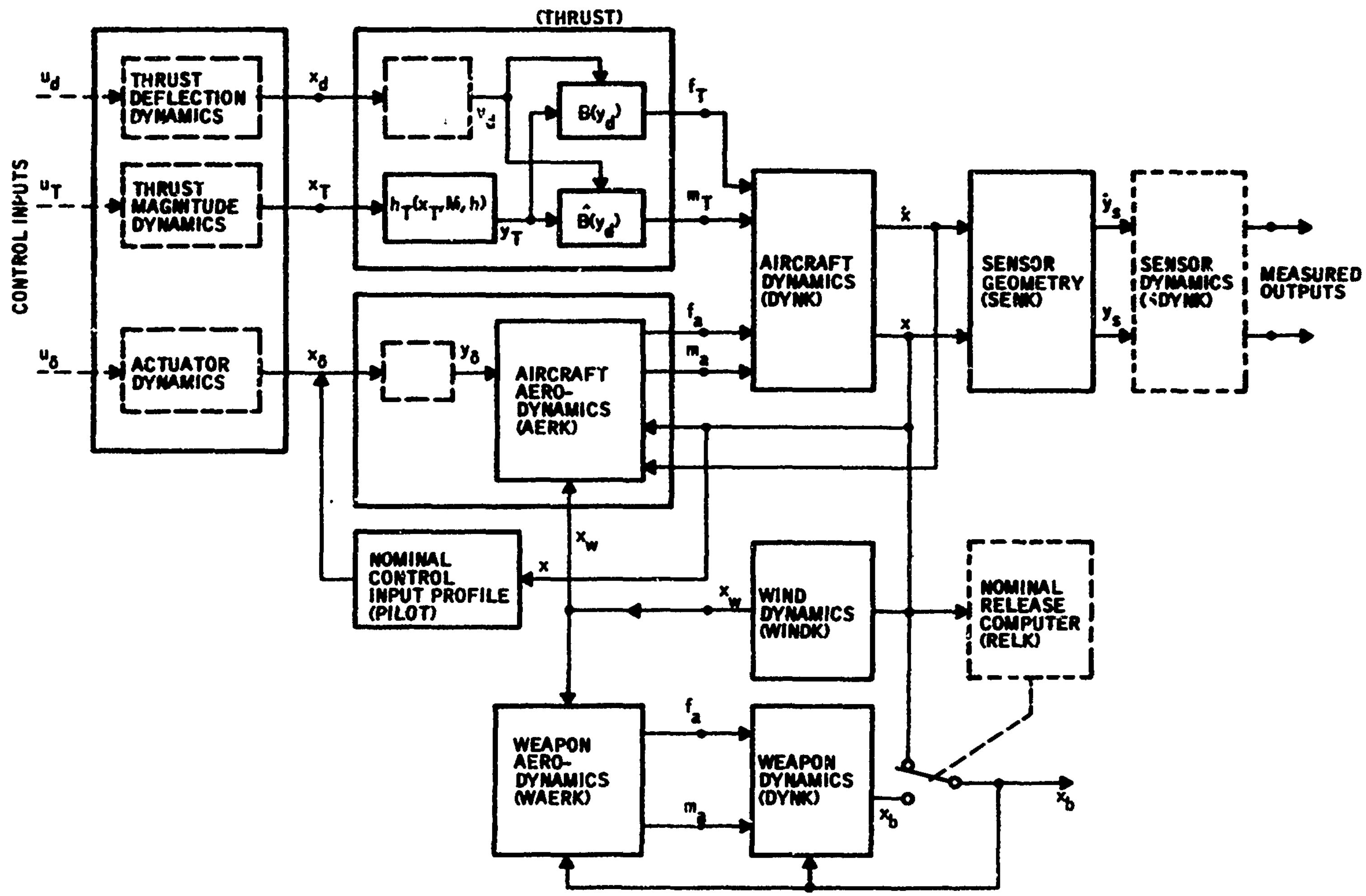


Figure 7. Information Flow in ADAP 1 System - Simulation Only (dotted blocks denote future implementations)

Table II. ADAP 1 A-Array Map (1-100)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
1	t	TM		26	α			51	$\cos\phi$	CPHI		76			
2	x_e	X, X4(1)		27	β			52	$\sin\psi$	SPSI		77	L	L	
3	y_e	Y, X4(2)		28	γ	GAM		53	$\cos\psi$	CPSI		78	M	M	
4	h	H, X4(3)		29	α_a	AL		54	e_{11}	E(1, 1)		79	N	N	
5	Ma	MACH		30	β_a	BET		55	e_{21}	E(2, 1)		80	p_a	PA	
6	a	SOS		31	θ	TH, X3(1)		56	e_{31}	E(3, 1)		81	q_a	QA	
7	u	U, X1(1)		32	ϕ	PHI, X3(2)		57	e_{12}	E(1, 2)		82	r_a	RA	
8	v	V, X1(2)		33	ψ	PSI, X3(3)		58	e_{22}	E(2, 2)		83	I_x	IX	
9	w	W, X1(3)		34	---	---		59	e_{32}	E(3, 2)		84	I_y	IY	
10	u_a	UA		35	---	---		60	e_{13}	E(1, 3)		85	I_z	IZ	
11	v_a	VA		36	$\dot{\gamma}$	GAMDOT		61	e_{23}	E(2, 3)		86	I_{xy}	IXY	
12	w_a	WA		37	$\dot{\alpha}_a$	ALDOT		62	e_{33}	E(3, 3)		87	I_{yz}	IYZ	
13	$\dot{x}_e$	XDOT, X4D(1)		38	$\dot{\beta}_a$	BETDOT		63	a_{x2}			88	I_{xz}	IXZ	
14	$\dot{y}_e$	YDOT, X4D(2)		39	$\dot{\theta}$	THDOT, X3D(1)		64	a_{y2}			89	Δt	DELT	
15	$\dot{h}$	HDOT, X4D(3)		40	$\dot{\phi}$	PHIDOT, X3D(2)		65	a_{z2}			90	g	G	
16	V_a	VEL		41	$\dot{\psi}$	PSIDOT, X3D(3)		66	χ	CHI		91	m	MASS	
17	$\dot{u}$	UDOT, X1D(1)		42	p	P, X2(1)		67	Δx_1			92	Δx_T	DXEX	
18	$\dot{v}$	VDOT, X1D(2)		43	q	Q, X2(2)		68	Δy_1			93	Δz_T	DZEX	
19	$\dot{w}$	WDOT, X1D(3)		44	r	R, X2(3)		69	Δz_1			94	Flag	FLG	
20	a_x	ACGX		45	$\dot{p}$	PDOT, X2D(1)		70	$\bar{q}$	QBAR		95	Δx_{ca}	DXCA	
21	a_y	ACGY		46	$\dot{q}$	QDOT, X2D(2)		71	Fx	FX		96	Δy_{ca}	DYCA	
22	a_z	ACGZ		47	$\dot{r}$	RDOT, X2D(3)		72	Fy	FY		97	Δz_{ca}	DZCA	
23	a_{x1}			48	$\sin\theta$	STH		73	Fz	FZ		98	Δx_3		
24	a_{x2}			49	$\cos\theta$	CTH		74	$\Delta\alpha$	DELA		99	Δy_2		
25	a_{x3}			50	$\sin\phi$	SPHI		75	$\Delta\delta$	DELD		100	Δz_2		

Table II. ADAP 1 A-Array Map (101-200)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
101	u_g	UG		126				151		LS		176			
102	v_g	VG		127				152		MS		177			
103	w_g	WG		128	q_c			153		NS		178			
104	p_g	PG		129	P_s			154		FLG		179			
105	q_g	QG		130		CONF		155		FBC		180			
106	r_g	RG		131	N	N		156	h_1	H1		181			
107	h_{min}	HMIN		132	ΔT			157	$\Delta\delta_{s1}$	DELDS1		182			
108	$\dot{h}$-1	HDOT1		133				158				183			
109				134		DTOUT		159				184			
110				135		TEOJ		160				185			
111	T_x	XT		136	---	---		161	λ_o	W1		186			
112	T_z	ZT		137	---	---		162	λ_1	W2		187			
113	T_y	YT		138		RUN		163	λ_2	W3		188			
114	ρ	RHO		139		OUT		164	λ_3	W4		189			
115	b	B		140	δs_{pu}	YDSPU		165	$\dot{\lambda}_o$	W1DOT		190			
116	c	C		141		YTH(1)		166	$\dot{\lambda}_1$	W2DOT		191			
117	S	S		142		YTH(2)		167	$\dot{\lambda}_2$	W3DOT		192			
118				143				168	$\dot{\lambda}_3$	W4DOT		193			
119	η			144				169	---			194			
120	P_T			145		N1		170	n_x	ANX, NX		195			
121	δ_s	YDS		146		N2		171	n_u	ANU, NU		196			
122	$\Delta\delta_s$	DELDS		147		TAPE UNIT		172	n_w	ANW, NW		197			
123	δ_a	YDA		148		LW, FXS		173				198			
124	δ_{ao}			149		FYS		174				199			
125	δ_r	YDR		150		FZS		175				200			

Table II. ADAP 1 A-Array Map (201-300)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
201	δu	DX(1)		226				251				276	t_p	TP	
202	δv	DX(2)		227				252				277	t_1	T1	
203	δw	DX(3)		228				253				278	t_{pu}	TPU	
204	δp	DX(4)		229				254				279	t_t	TT	
205	δq	DX(5)		230				255				280	t_r	TR	
206	δr	DX(6)		231	δua	DW(1)		256	$K_{\dot{q}}$	KQDOT		281	t_f	TF	
207	$\delta \theta$	DX(7)		232	δva	DW(2)		257	K_q	KQ		282	t_d	TD	
208	$\delta \phi$	DX(8)		233	δwa	DW(3)		258	$K_{\dot{\gamma}}$	KGAMDT		283	t_{max}	TMAX	
209	$\delta \psi$	DX(9)		234	δpa	DW(4)		259	K_γ	KGAM		284	R_p	RP	
210	δxe	DX(10)		235	δqa	DW(5)		260	γ_o	GAMN		285	ΔV_e	DELVE	
211	δye	DX(11)		236	δra	DW(6)		261		TINT		286	N	CN	
212	δhe	DX(12)		237				262		TFLG		287	Δt_t	DTT	
213	$\delta \dot{u}$			238				263	δx	DELTAZ		288	---	---	
214	$\delta \dot{v}$			239				264	δy	DELTAY		289	$\bar{V}$	VELB	
215	$\delta \dot{w}$			240				265	$\Delta t/4$	DELT4		290	$\bar{\psi}$	PSIB	
216	$\delta \dot{p}$			241				266	$\Delta t/2$	DHDLT		291	$\bar{\theta}$	THETAB	
217	$\delta \dot{q}$			242				267		HMINS		292	h_o	HNAUT	
218	$\delta \dot{r}$			243				268		ISTRT		293	$\bar{V}_o$	VBNAUT	
219				244				269		IEND		294	h_d	HMBD	
220				245				270	q_{pu}	QPU		295	$\bar{u}$	UB	
221	$\delta(\delta_a)$	DU(1)		246				271	t_{test}	TTEST		296	$\bar{v}$	VB	
222	$\delta(\delta_s)$	DU(2)		247				272	Δt_{last}	DELTTL		297	$\bar{w}$	WB	
223	$\delta(\delta_r)$	DU(3)		248				273	Δt_{TRIM}	DELTTRM		298	$\bar{u}_a$	UBA	
224	$\delta(\delta_{sp})$	DU(4)		249				274	h_p	HP		299	$\bar{v}_a$	VBA	
225				250				275	t_o	TN		300	$\bar{w}_a$	WBA	

Table II. ADAP 1 A-Array Map (301-400)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
301	$\bar{V}_a$	VBARA		326		YRPS(1)		351	$C_{x\alpha}$			376	C_{MM}		
302	σ_u	SIGU		327	y_{RPS}	YRPS(2)		352	$C_{x\dot{\alpha}}$			377	$C_{M\alpha}$		
303	σ_v	SIGV		328		YRPS(3)		353	C_{xq}			378	$C_{M\dot{\alpha}}$		
304	σ_w	SIGW		329		YRVS(1)		354	$C_{x\delta ei}$			379	C_{Mq}		
305	a_u	AU		330	y_{RVS}	YRVS(2)		355	$C_{x\delta eo}$			380	$C_{M\delta ei}$		
306	a_v	AV		331		YRVS(3)		356	$C_{y\beta}$			381	$C_{M\delta eo}$		
307	a_w	AW		332				357	$C_{y\dot{\beta}}$			382	$C_{N\beta}$		
308	a_p	AP		333				358	C_{yp}			383	$C_{N\dot{\beta}}$		
309	a_q	AQ		334				359	C_{yr}			384	C_{Np}		
310	a_r	AR		335				360	$C_{y\delta ai}$			385	C_{Nr}		
311		Y1S(1)		336				361	$C_{y\delta ao}$			386	$C_{N\delta ai}$		
312	y_{1S}	Y1S(2)		337				362	$C_{y\delta r}$			387	$C_{N\delta ao}$		
313		Y1S(3)		338				363	C_{zm}			388	$C_{N\delta r}$		
314		Y2S(1)		339				364	$C_{z\alpha}$			389	---		
315	y_{2S}	Y2S(2)		340				365	$C_{z\dot{\alpha}}$			390	---		
316		Y2S(3)		341				366	C_{zq}			391	e_{a11}	EA(1, 1)	
317		Y3S(1)		342				367	$C_{z\delta ei}$			392	e_{a21}	EA(2, 1)	
318	y_{3S}	Y3S(2)		343				368	$C_{z\delta eo}$			393	e_{a31}	EA(3, 1)	
319		Y3S(3)		344				369	$C_{L\beta}$			394	e_{a12}	EA(1, 2)	
320		Y1SD(1)		345				370	$C_{L\dot{\beta}}$			395	e_{a22}	EA(2, 2)	
321	$\dot{y}_{1S}$	Y1SD(2)		346				371	C_{Lp}			396	e_{a32}	EA(3, 2)	
322		Y1SD(3)		347				372	C_{Lr}			397	e_{a13}	EA(1, 3)	
323		Y2SD(1)		348				373	$C_{L\delta ai}$			398	e_{a23}	EA(2, 3)	
324	$\dot{y}_{2S}$	Y2SD(2)		349				374	$C_{L\delta ao}$			399	e_{a33}	EA(3, 3)	
325		Y2SD(3)		350				375	$C_{L\delta r}$			400	e_{v11}	EV(1, 1)	

Table II. ADAP 1 A-Array Map (401-500)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
401	e_{v21}	EV(2, 1)		426	$e_{\dot{\omega}33}$	EOMEGD(3, 3)		451				476			
402	e_{v31}	EV(3, 1)		427	$e_{\omega 11}$	EOMEG(1, 1)		452				477			
403	e_{v12}	EV(1, 2)		428	$e_{\omega 21}$	EOMEG(2, 1)		453				478			
404	e_{v22}	EV(2, 2)		429	$e_{\omega 31}$	EOMEG(3, 1)		454				479			
405	e_{v32}	EV(3, 2)		430	$e_{\omega 12}$	EOMEG(1, 2)		455				480			
406	e_{v13}	EV(1, 3)		431	$e_{\omega 22}$	EOMEG(2, 2)		456				481			
407	e_{v23}	EV(2, 3)		432	$e_{\omega 32}$	EOMEG(3, 2)		457				482			
408	e_{v33}	EV(3, 3)		433	$e_{\omega 13}$	EOMEG(1, 3)		458				483			
409	$e_{\Omega 11}$	ECOMEG(1, 1)		434	$e_{\omega 23}$	EOMEG(2, 3)		459				484			
410	$e_{\Omega 12}$	ECOMEG(2, 1)		435	$e_{\omega 33}$	EOMEG(3, 3)		460				485			
411	$e_{\Omega 31}$	ECOMEG(3, 1)		436	Δx_a	DRA(1)		461				486			
412	$e_{\Omega 12}$	ECOMEG(1, 2)		437	Δy_a			462				487			
413	$e_{\Omega 22}$	ECOMEG(2, 2)		438	Δz_a			463				488			
414	$e_{\Omega 32}$	ECOMEG(3, 2)		439	Δx_v	DRV(1)		464				489			
415	$e_{\Omega 13}$	ECOMEG(1, 3)		440	Δy_v			465				490			
416	$e_{\Omega 23}$	ECOMEG(2, 3)		441	Δz_v			466				491			
417	$e_{\Omega 33}$	ECOMEG(3, 3)		442	Δx_R	DRR(1)		467				492			
418	$e_{\dot{\omega}11}$	EOMEGD(1, 1)		443	Δy_R			468				493			
419	$e_{\dot{\omega}21}$	EOMEGD(2, 1)		444	Δz_R			469				494			
420	$e_{\dot{\omega}31}$	EOMEGD(3, 1)		345				470				495			
421	$e_{\dot{\omega}12}$	EOMEGD(1, 2)		446				471				496			
422	$e_{\dot{\omega}22}$	EOMEGD(2, 2)		447				472				497			
423	$e_{\dot{\omega}32}$	EOMEGD(3, 2)		448				473				498			
424	$e_{\dot{\omega}13}$	EOMEGD(1, 3)		449				474				497			
425	$e_{\dot{\omega}23}$	EOMEGD(2, 3)		450				475				500			

Table II. ADAP 1 A-Array Map (501-600)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
501		XEX(1)		526		YTB(1, 5)		551		E2(2, 2)		576		YTH(3)	
502		XEX(2)		527		YTB(2, 5)		552		E2(1, 3)		577		YTH(4)	
503	x_{ex}	XEX(3)		528		CTH(1, 1)		553		E2(2, 3)		578		YTH(5)	
504		XEX(4)		529		CTH(2, 1)		554		E2(1, 4)		579		XCA	
505		XEX(5)		530		CTH(1, 2)		555		E2(2, 4)		580		YCA	
506		YEX(1)		531		CTH(2, 2)		556		E2(1, 5)		581		ZCA	
507		YEX(2)		532		CTH(1, 3)		557		E2(2, 5)		582		YAB(1)	
508	y_{ex}	YEX(3)		533		CTH(2, 3)		558		EÔ		583		YAB(2)	
509		YEX(4)		534		CTH(1, 4)		559		XTH(1)		584		YAB(3)	
510		YEX(5)		535		CTH(2, 4)		560		XTH(2)		585		YAB(4)	
511		ZEX(1)		536		CTH(1, 5)		561		XTH(3)		586		YAB(5)	
512		ZEX(2)		537		CTH(2, 5)		562		XTH(4)		587		YEB(1)	
513	z_{ex}	ZEX(3)		538		E1(1, 1)		563		XTH(5)		588		YEB(2)	
514		ZEX(4)		539		E1(2, 1)		564		XAD(1)		589		YEB(3)	
515		ZEX(5)		540		E1(1, 2)		565		XAD(2)		590		YEB(4)	
516	n_{td}	ANTD		541		E1(2, 2)		566		XAD(3)		591		YEB(5)	
517	n_{ex}	ANEX		542		E1(1, 3)		567		XAD(4)		592			
518		YTB(1, 1)		543		E1(2, 3)		568		XAD(5)		593		B(1, 1)	
519		YTB(2, 1)		544		E1(1, 4)		569		XED(1)		594		B(3, 1)	
520		YTB(1, 2)		545		E1(2, 4)		570		XED(2)		595		BH(2, 1)	
521		YTB(2, 2)		546		E1(1, 5)		571		XED(3)		596	C_1	B(1, 2)	
522		YTB(1, 3)		547		E1(2, 5)		572		XED(4)		597	C_2	B(3, 2)	
523		YTB(2, 3)		548		E2(1, 1)		573		XED(5)		598	C_3	BH(2, 2)	
524		YTB(1, 4)		549		E2(2, 1)		574		YTH(1)		599			
525		YTB(2, 4)		550		E2(1, 2)		575		YTH(2)		600		RLT	

Table II. ADAP 1 A-Array Map (601-700)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
601		RMT		626				651	Y_{TRM}			676			
602		RNT		627				652				677	$\$T_{h1}$		
603	h_x			628				653				678	$\$T_{h2}$		
604	h_y			629				654				679			
605	h_z			630				655				680			
606	T_{h1}	T(1), U(7)		631				656				681			
607	T_{h2}	T(2), U(8)		632				657				682			
608	T_{c1}	TC1		633				658				683			
609	T_{c2}	TC2		634				659				684			
610	τ_{ecg}			635				660				685			
611	x_{cg}	XCG		636				661	V_{knot}			686			
612	z_{cg}	ZCG		637				662				687			
613				638				663				688			
614				639				664				689			
615				640				665				690			
616				641				666				691			
617				642				667				692			
618				643				668				693			
619				644				669				694			
620				645	δ_{sp}	U(4)		670				695			
621				646	δ_{sb}	U(5)		671				696			
622				647	δ_{lp}	U(6)		672				697			
623				648	δ_{lg}	U(7)		673				698			
624				649				674				699			
625				650				675				700	θ^*	THD	

Table II. ADAP 1 A-Array Map (701-800)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
701	φ	PHID		726		DPTV		751			0.0 or 99999	776			
702	ψ^*	PSID		727				752				777			
703	γ^*	GAMD		728				753				778			
704	R	RDOTD		729				754				779			
705	q^*	QDOTD		730				755				780			
706	α^*	ALD		731				756				781			
707	β^*	BETD		732				757				782			
708	$\dot{\gamma}$	GAMDTD		733				758				783			
709	$\dot{\alpha}^*$	ALDOTD		734				759				784			
710	$\dot{\beta}^*$	BETDTD		735				760				785			
711	p^*	PD		736				761				786			
712	q^*	QD		737				762				787			
713	r^*	RD		738				763				788			
714	χ^*	CHID		739				764				789			
715	$\dot{\nu}^*$			740				765				790			
716	wt	WT		741				766				791			
717	K_γ	KGAM		742				767				792			
718				743				768				793			
719				744				769				794			
720	ξ_1			745				770				795			
721	η_1			746				771				796			
722	ξ_2			747				772				797			
723	η_2			748				773				798			
724	η_3	FLOAD		749				774				799			
725	V_T			750				775				800			

Table II. ADAP 1 A-Array Map (801-900)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
801	C_L	CL, F(1)		826	$C_{n\beta}$	CNBET, F(26)		851				876	$C_{N\delta}$	CNDEL, F(76)	
802	C_{zq}	CZQ, F(2)		827	C_{nr}	CNR, F(27)		852				877	C_A	CA, F(77)	
803	$C_{z\dot{\alpha}}$	CZALDT, F(3)		828	C_{np}	CNP, F(28)		853	ρ	RHO, F(53)		878	C_m	CM, F(78)	
804	$C_{L\delta s}$	CLDS, F(4)		829	$C_{n\delta SP}$	CNDSP, F(29)		854	a	SOS, F(54)		879	C_{mq}	CMQ, F(79)	
805	$C_{L\delta SP}$	CLDSP, F(5)		830	$C_{n\delta a}$	CNDA, F(30)		855				880	$C_{m\delta}$	CMDEL, F(80)	
806	$C_{L\delta a}$	CLDA, F(6)		831	$C_{n\delta r}$	CNDR, F(31)		856				881			
807	$C_{L\delta SB}$	CLDSB, F(7)		832	$C_{\ell\beta}$	CLLBET, F(32)		857				882			
808	$C_{L\delta\ell g}$	CLDLG, F(8)		833	$C_{\ell r}$	CLLR, F(32)		858				883			
809	C_D	CD, F(9)		834	$C_{\ell p}$	CLLP, F(34)		859				884			
810	$C_{(D\delta SB)}$	CDDSB, F(10)		835	$C_{\ell\delta SP}$	CLLDSP, F(35)		860				885			
811	$C_{D\delta\ell g}$	CDDLG, F(11)		836	$C_{\ell\delta a}$	CLLDA, F(36)		861				886			
812	C_{mca}	CMCA, F(12)		837	$C_{\ell\delta r}$	CLLDR, F(37)		862				887			
813	C_{mq}	CMQ, F(13)		838				863				888			
814	$C_{m\dot{\alpha}}$	CMALDT, F(14)		839				864				889			
815	$C_{m\delta s}$	CMDS, F(15)		840				865				890			
816	$C_{m\delta SP}$	CMDSP, F(16)		841				866				891			
817	$C_{m\delta a}$	CMDA, F(17)		842				867				892			
818	$C_{m\delta SB}$	CMDSB, F(18)		843				868				893			
819	$C_{m\delta\ell g}$	CMDLG, F(18)		844				869				894			
820	$C_{y\beta}$	CYBET, F(20)		845	X_{cg}	XCG, F(45)		870				895			
821	C_{yr}	CYR, F(21)		846	Z_{cg}	ZCG, F(46)		871				896			
822	C_{yp}	CYP, F(22)		847	I_x	IX, F(47)		872				897			
823	$C_{y\delta SP}$	CYDSP, F(23)		848	I_y	IY, F(48)		873				898			
824	$C_{y\delta a}$	CYDA, F(24)		849	I_z	IZ, F(49)		874				899			
825	$C_{y\delta r}$	CYDR, F(25)		850	I_{xz}	IXZ, F(50)		875	C_N	CN, F(75)		900			

Table II. ADAP 1 A-Array Map (901-1000)

A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value	A( )	Symbol	Mnemonic	Value
901	u(loc)	AI(1)		926	$\delta\delta$(loc)	AI(26)		951				976			
902	v(loc)	AI(2)		927	δr(loc)	AI(27)		952				977			
903	w(loc)	AI(3)		928	δ_{SP}(loc)	AI(28)		953				978			
904	p(loc)	AI(4)		929	u_g(loc)	AI(29)		954				979			
905	q(loc)	AI(5)		930	v_g(loc)	AI(30)		955				980			
906	r(loc)	AI(6)		931	w_g(loc)	AI(31)		956				981			
907	θ(loc)	AI(7)		932	p_g(loc)	AI(32)		957				982			
908	ϕ(loc)	AI(8)		933	q_g(loc)	AI(33)		958				983			
909	ψ(loc)	AI(9)		934	r_g(loc)	AI(34)		959				984			
910	x_e(loc)	AI(10)		935				960				985			
911	y_e(loc)	AI(11)						961				986			
912	h_e(loc)	AI(12)						962				987			
913	$\dot{u}$(loc)	AI(13)						963				988			
914	$\dot{v}$(loc)	AI(14)						964				989			
915	$\dot{w}$(loc)	AI(15)		940				965				990			
916	$\dot{p}$(loc)	AI(16)						966				991			
917	$\dot{q}$(loc)	AI(17)						967				992			
918	$\dot{r}$(loc)	AI(18)						968				993			
919	$\dot{\theta}$(loc)	AI(19)						969				994			
920	$\dot{\phi}$(loc)	AI(20)		945				970				995		SAC	
921	$\dot{\psi}$(loc)	AI(21)						971				996		SW	
922	$\dot{x}_e$(loc)	AI(22)						972				997		TABRD	
923	$\dot{y}_e$(loc)	AI(23)						973				998		ACWS	
924	$\dot{h}_e$(loc)	AI(24)						974				999			
925	δ_a(loc)	AI(25)		950				975				1000			

In the A-Array Map Form (Table II), the value column is auxiliary and can be used to record the input values of a particular simulation run. For programming convenience, A-array locations are equivalenced to mnemonic names. For example, if the variable α is to be assigned to location 26 in the A-array, then either a statement

```
EQUIVALENCE (ALPHA, A(026))
```

is used at the beginning of a subroutine or a statement

```
ALPHA = A(26)
```

is used within the subroutine. Each subroutine begins with an equivalence block. The equivalence block is divided into three parts. The "parameter inputs" are those inputs to the subroutine which are provided through ADAP's main input. The "variable inputs" are those inputs which are computed in some other subroutine in the ADAP system. The "variable outputs" are those quantities which are computed in the subroutine.

INPUT DESCRIPTION

Input for the ADAP 1 is in the form of punched cards. It is divided into two groups:

- Common card data input
- Table card data input

Both groups are read in during the first call (Mode = -1) of subroutine EXEK.

Common Card Data Input

The common input is the input that appears after the data control card behind the program deck. This input consists of:

- Input comment cards
- A-array input cards
- A-array output specification cards
- Program control cards

Input Comment Cards -- Any number of comment cards may be used in the common input deck. Although these are not required, they are useful in defining each simulation and linearization case. Each comment card must have a C in Column 1 as shown in Figure 8. Columns 2-80 of each card are printed on the first page of output.

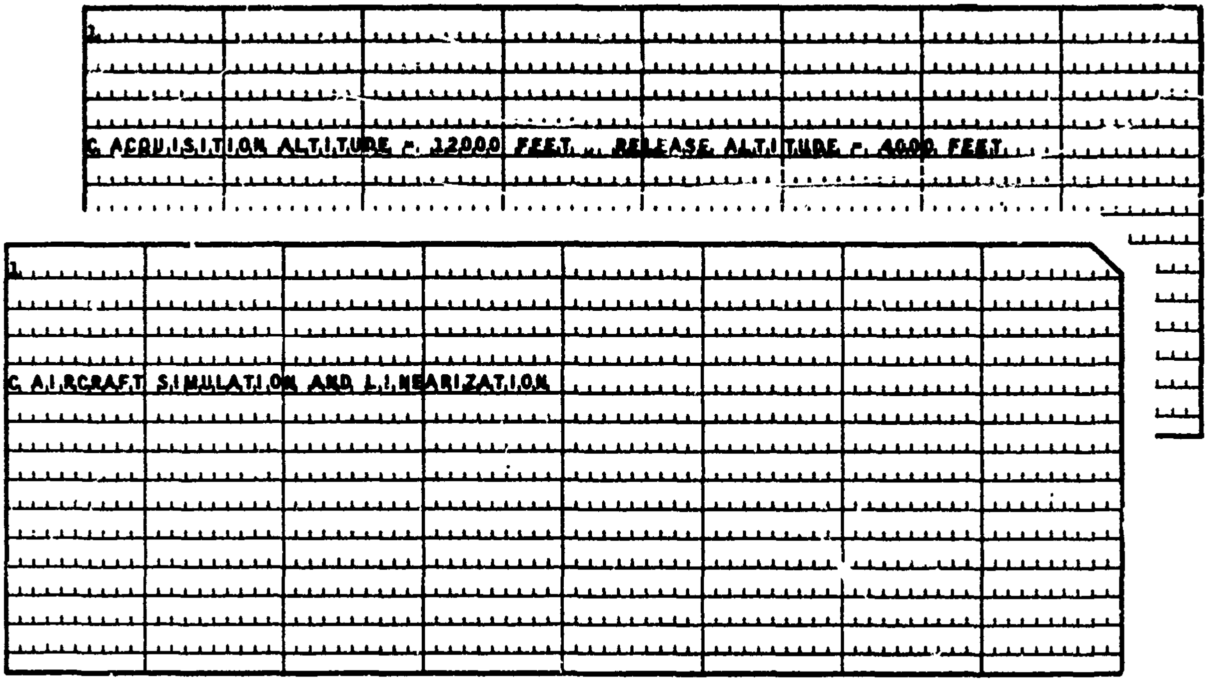


Figure 8. Input Comment Cards

A-Array Input Cards -- It is possible to assign a value to any location in the A-array through input cards. This is done by a control card PCC starting in column 1, followed by a set of cards containing the array numbers and corresponding A-array values (Figure 9). From one to five A-array locations can be input by a single card.

The format for this card is shown in Table III.

Figure 9 shows five values punched on a single card. This method should be used for large amounts of input data. A single card may also be used for each A-array value input. This method is satisfactory when the number of inputs are small, since it simplifies correcting and/or modifying an input card. There is no limit on the number of input cards that can be used.

If a format data mismatch occurs while reading an A-array input card, then a message, "CARD ERROR" will be printed, followed by a printout of the card in error. The program will stop on a format data mismatch.

A-Array Output Specification Cards -- The A-array locations that must be printed out during a run are specified by a control card PRINT starting column 1, followed by a set of cards listing the indices of the A-arrays that will be printed (Figure 10). These numbers are right justified in I_5 fields on a print card. The format for this card is illustrated in Table IV.

Table III. A-Array Input Card Format

Columns	Description
1 - 4	A-array index I_1
5 - 15	Value of $A(I_1)$
16 - 19	A-array index I_2
20 - 30	Value of $A(I_2)$
31 - 34	A-array index I_3
35 - 45	Value of $A(I_3)$
46 - 49	A-array index I_4
50 - 60	Value of $A(I_4)$
61 - 64	A-array index I_5
65 - 75	Value of $A(I_5)$

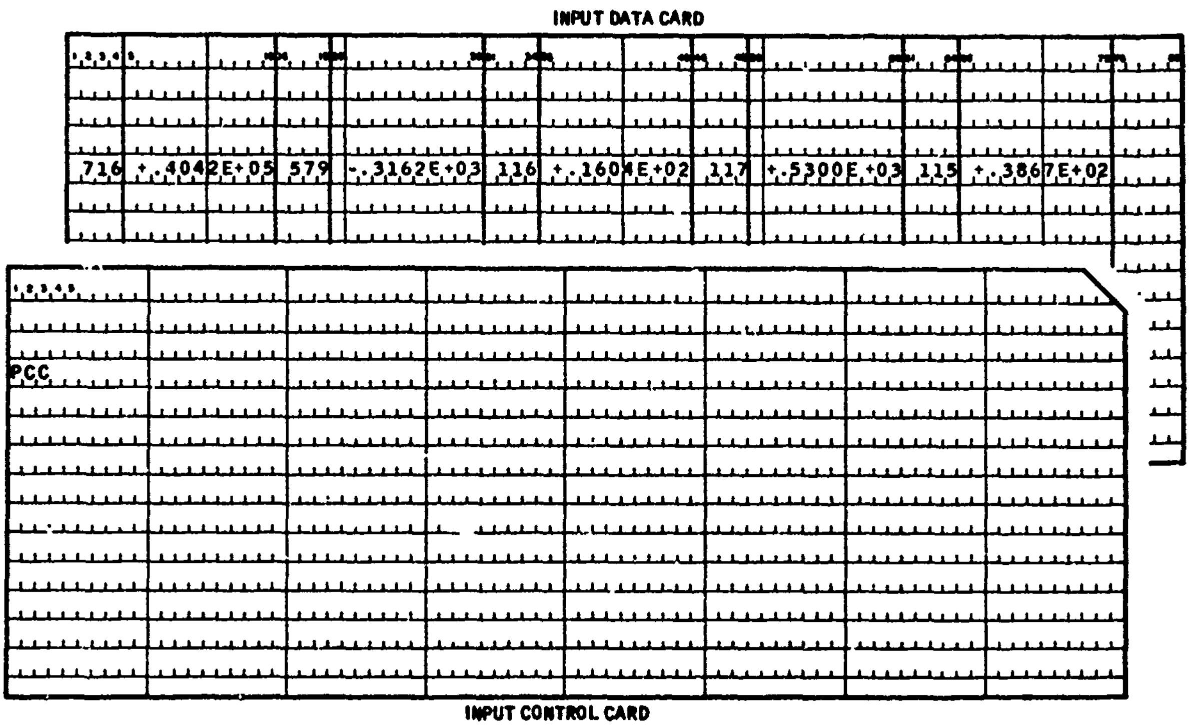


Figure 9. A-Array Input Cards

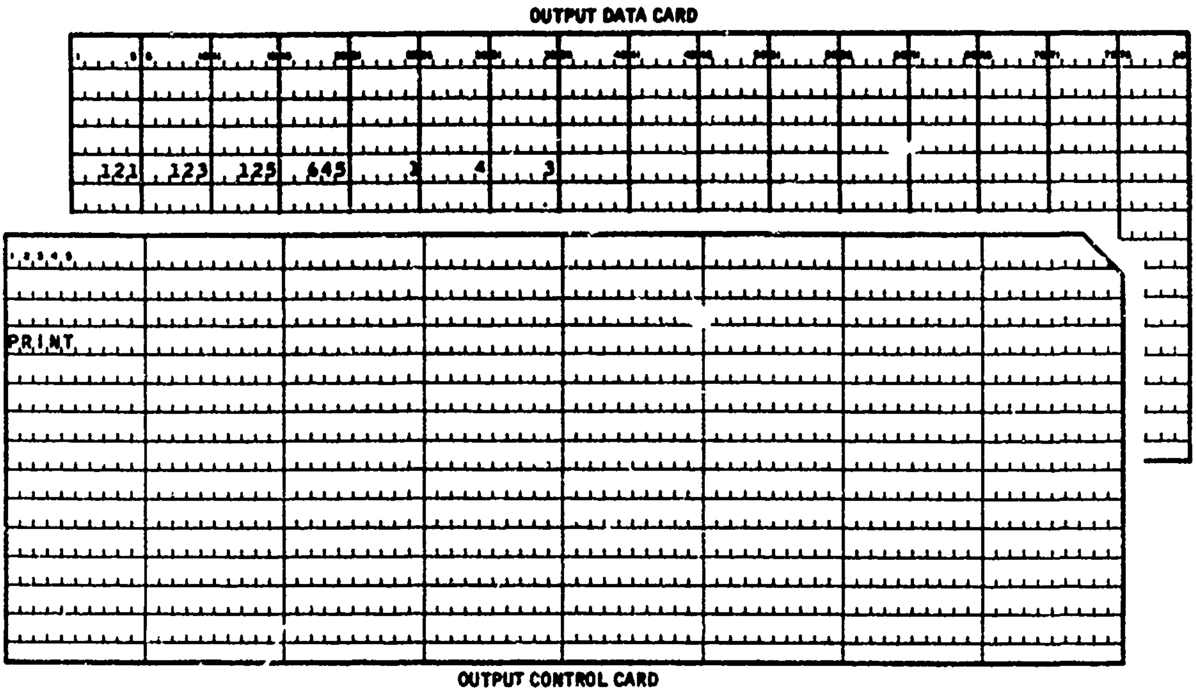


Figure 10. A-Array Output Cards

Table IV. A-Array Output Card Format

Columns	Description
1 - 5 6 - 10 11 - 15 16 - 20 21 - 25 26 - 30 31 - 35 36 - 40 41 - 45 46 - 50 51 - 55 56 - 60 61 - 65 66 - 70 71 - 75 76 - 80	Indices of A-arrays to be printed out during a simulation

All blank fields are ignored. Any number of cards may be used, but the number of A-array locations specified for printing must be less than 100.

If a format data mismatch occurs while reading a print index card then a message, "PRINT SPEC ERROR," will be printed, followed by a printout of the card in error. The program will stop on a format data mismatch.

<u>Program Control Cards</u> -- Two program control cards are used in the input:

- RUN
- STOP

Each of these control words must start in column 1 (Figure 11). The RUN card signifies the end of common input for one case. When the program reads the RUN card, it will begin to execute.

The STOP card is the last card in a data deck. It commands the program to stop.

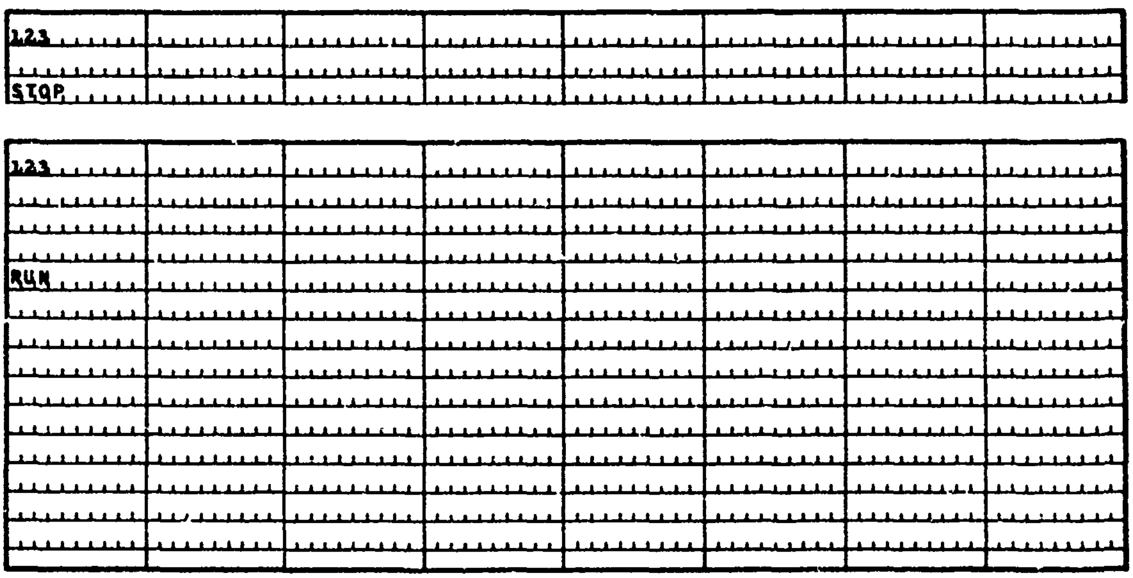


Figure 11. Program Control Cards

<u>Rules for Common Card Data Input</u> -- The following rules apply to common card data input:

- Comment cards must have a C in column 1.
- Any number of comment cards may be used, including zero.
- The PRINT and PCC sections can appear in any order.
- All integers must be right-justified in their input fields.

Table Card Data Input

The table card data are placed after the RUN card in the input data deck. The input cards required to specify a function table are:

- Function header card
- Variable-value cards
- Function value cards
- End function card

These cards are shown in Figure 12.

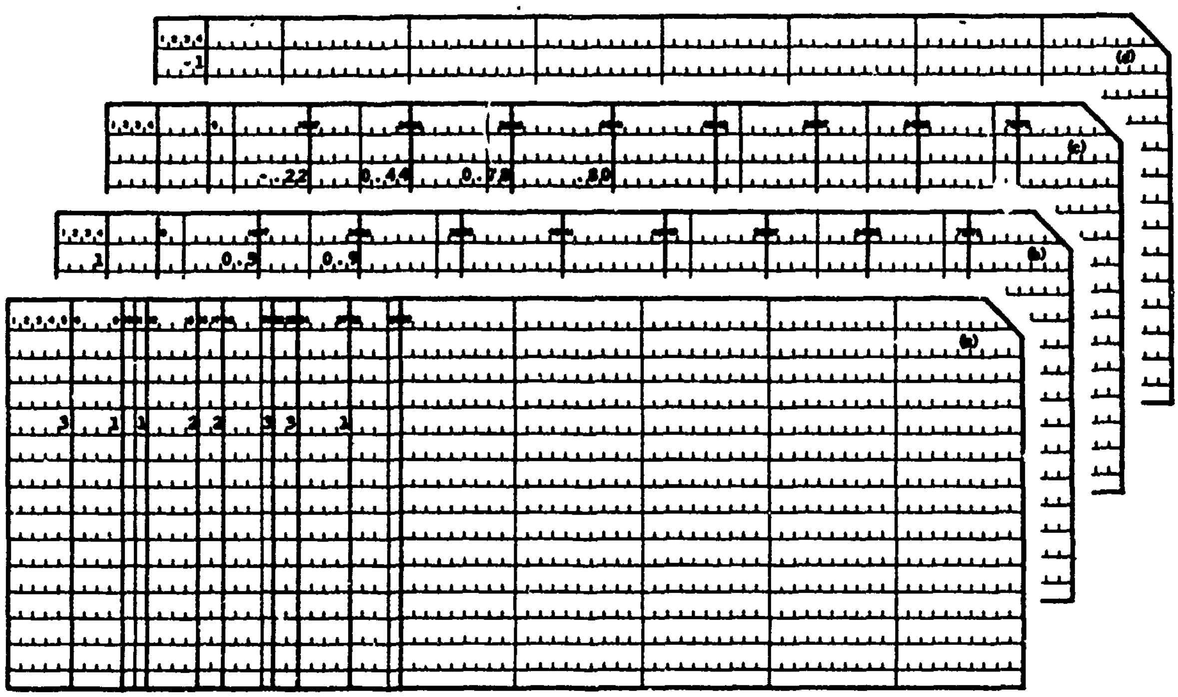


Figure 12. Table Input Cards: (a) Function Header Card; (b) Variable-Value Cards; (c) Function Value Cards; (d) End Function Card

Function Header Card -- The following information is contained on a function header card:

- Number of variables in the table
- The integers assigned to the functions variables
- The integers assigned to variable-value sets used in the table
- The integer assigned to the function
- The integer assigned to a function which has exactly the same table of function values as the function at hand (provided such a function exists)

The format for this card is shown in Table V.

Table V. Function Header Card Format

Columns	Description	
1 - 5	Number of variables	
6 - 9		First variable value set
10 - 11		First variable
12 - 15		Second variable value set
16 - 17		Second variable
18 - 21	Integer assigned to	Third variable value set
22 - 23		Third variable
24 - 27		Function
28 - 31		Function with same table

All entries on this card are integers and must be right-jusitfied; i. e., the least significant digit must be in the right-most column of the field (Figure 12).

Variable-Value Cards -- A variable-value set is a set of numbers that a variable takes on in a function table. In other words, these numbers are the variable values at which function values are given in the table.

The variable values in a set are specified in the input immediately after the function header card (Figure 12). The set number associated with the entries on a card must be specified in columns 1-4. From one to nine values of a variable can be specified on one card. Beginning in column 9, a new field starts every eight columns. The format for these cards is shown in Table VI.

If more than one card is used to enter a set, then each card must be identified by the appropriate set number. Blank variable-value fields will be ignored; therefore, a zero value must be explicitly denoted as 0.0. The first time a variable-value set is referenced on a function header card, the values in that set must be specified immediately after the function header card. However, on subsequent references to the set, it is only necessary to write the set number on the header card. It is not necessary to specify the numbers in the set again. Any sets that are specified must be specified in the same order as they appear on the function header card.

Table VI. Variable-Value Card Format

Columns	Description
1 - 4	Variable-value set number
9 - 16	1st
17 - 24	2nd
25 - 32	3rd
33 - 40	4th
41 - 48	5th Value of variable in the set
49 - 56	6th
57 - 64	7th
65 - 72	8th
73 - 80	9th

Function Value Cards -- Function values are specified under the same format as the variable values except the set number field; i. e., columns 1-4 are blank. Any card after the function header card which has no entry in columns 1-4 will be considered a function value card by the program. Function value cards must appear after the variable value cards (Figure 12). Blank entries will be ignored and zeros must be specified explicitly. The format for these cards is shown in Table VII.

Table VII. Function Value Card Format

Columns	Description
9 - 16	1st
17 - 24	2nd
25 - 32	3rd
33 - 40	4th
41 - 48	5th Value of function in table
49 - 56	6th
57 - 64	.
65 - 72	.
73 - 80	.

The order in which the function values are specified must correspond to the order in which the variables are specified. The function values are to be given with the last variable specified on the function header card varying fastest and the first varying slowest.

As an example consider the aerodynamic lift coefficient C_L (Ma, h, $\alpha°_w$) (Table VIII). The look-up representation of this table in ADAP 1 is F1(1,2,3) (see Subroutine AERK). This means that the function number is 1, the first variable number is 1, the second variable number is 2 and the third variable number is 3. The first variable-value set in the table is $M_1 = 0.5$ and $M_2 = 0.9$. The assigned set number is 1. The second variable value set in the table is $h_1 = 0.0$, and $h_2 = 20,000.0$. The assigned set number is 2. The third variable value set in the table is $\alpha_1 = -4.0$, $\alpha_2 = 8.0$, $\alpha_3 = 16.0$ and $\alpha_4 = 20.0$. The assigned set value is 3.

The function values are input as shown in Table VIII with the format given in Table VII. Figure 12(c) shows the first function value card in the deck.

Table VIII. $C_L(M, h, \alpha)$ Function Values

Variable Values		$\alpha_1 = -4.0$	$\alpha_2 = 8.0$	$\alpha_3 = 16.0$	$\alpha_4 = 20.0$	Function Value Cards
$M_1 = 0.5$	$h_1 = 0.0$	-0.22	0.44	0.78	0.80	1st
	$h_2 = 20,000.0$	-0.22	0.47	0.82	0.85	2nd
$M_2 = 0.9$	$h_1 = 0.0$	-0.25	0.46	0.74	0.80	3rd
	$h_2 = 20,000.0$	-0.27	0.51	0.82	0.89	4th

End Function Card -- The last data card for each function must be a -1 in columns 3 and 4. An extra end function card must be placed behind the last function in a deck. In other words, the last two cards in a function table data deck are -1's in columns 3 and 4.

General -- When choosing points on a curve that will be used to represent it in the program, there are several things to remember:

- The program interpolates linearly between stored points.
- The program does not extrapolate beyond stored points.
- Execution time is nearly independent of the number of points stored.

The entire ADAP 1 input data package is shown in Figure 13.

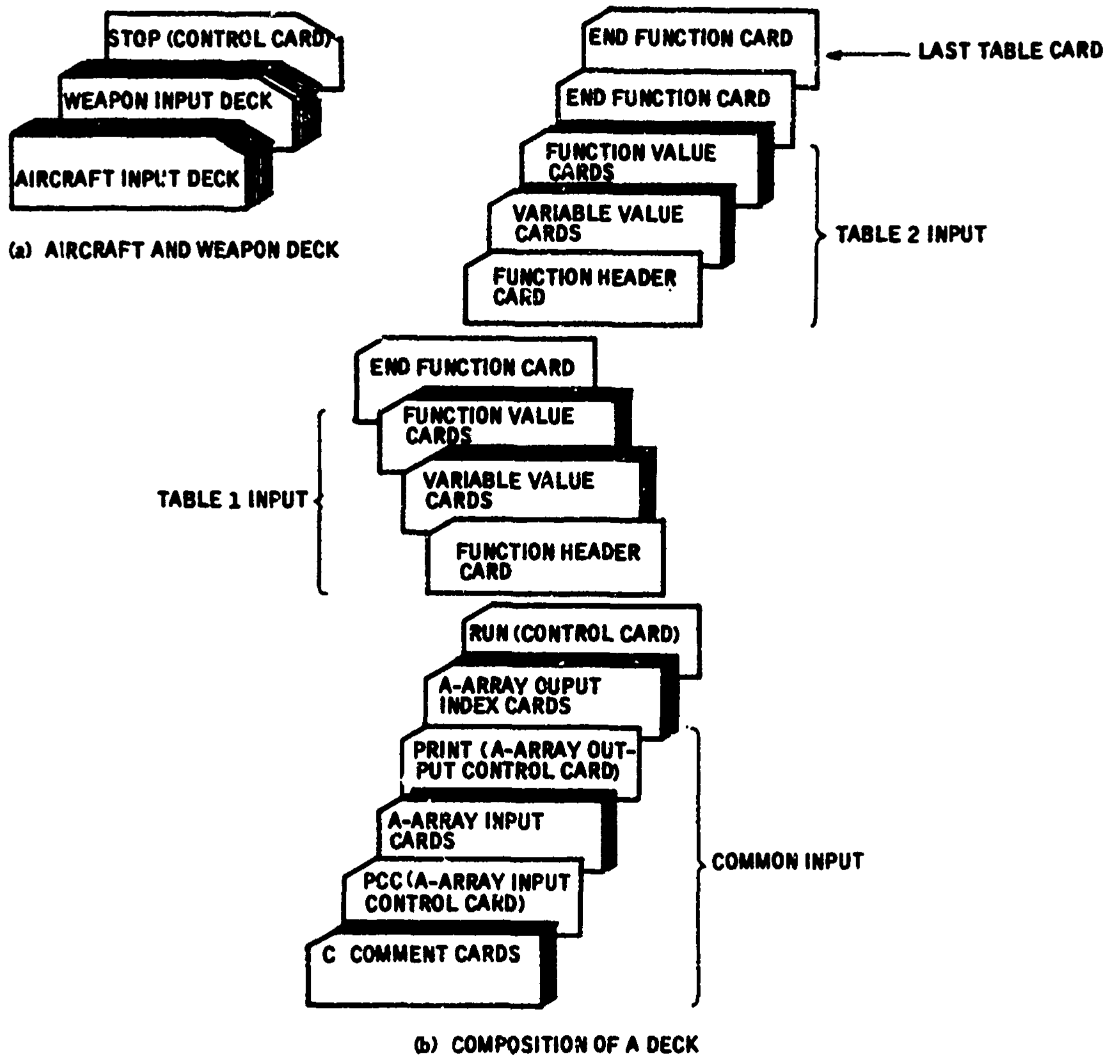


Figure 13. ADAP 1 Input Data Package

OUTPUT DESCRIPTION

The forms of output supplied with ADAP 1 are:

- Printed output of the input data deck
- Specified A-arrays printout
- Printed output of A-array dumps
- Linear data printout
- Linear data output to permanent disc file

Printed Output of Input Data Deck

Complete image of the input data deck is printed out at the beginning of each simulation and linearization run.

Specified A-Arrays Printout

The printed output of specified A-arrays is not used in the present program. The interval between output points is specified by DTOUT in A(134).

A-Array Dump

The term "A-array dump" means that the value of all 999 locations in the A-array are printed. (Only nonzero values are printed.) The interval between dump points is specified by DTOUT in A(134). This form of output is very useful for debugging and is used in ADAP 1. The value of time in seconds is always contained in A(1). The array location is printed first, followed by the value in that location.

Linear Data Printout

The matrices obtained in the linearizer are printed out by matrix print subroutine MP. Each matrix as well as its rows are identified. The documentation for MP is given in Section IV. The linear data generated during aircraft dive are F, G1, G2, HP, FW, GW and HW matrices. The linear data generated during weapon fall are F, G1, G2, FW, GW and HW matrices.

Linear Data Output to Permanent Disc Files

The linear data for aircraft and wind systems are augmented and output to a permanent disc file as the matrices FF, GG1, GG2, GG3, H2 and VW. The linear data output for aircraft occurs in subroutines LINK and SLINK. The linear data for weapon and wind systems are augmented and output to another permanent disc file as the matrices F and G3. This is done in subroutine WLINK.

ADAP 1 PROGRAM DESCRIPTION

ADAP 1 MAIN PROGRAM

All programs in ADAPS are written as subroutines. The main program ADAP 1 is used to tie these subroutines into a simulation and linearization. The main program flow diagram is shown in Figure 14 and the program listing in Figure 15.

The usage of the main program is self-explanatory from the figure. Because of its importance however a brief description is given below.

The main program can be divided into two sections -- initialization and simulation. In the initialization section, the A-Array is first cleared, the mode flag is set to -1, and subroutine EXEK is called. In this call, all input cards for one run are read by subroutines PREAD, PRINT and FLO[illegible]. Then the aircraft simulation flag SAC is tested. If SAC = 0, control is transferred to the weapon simulation (WS) part of the program. Otherwise all aircraft subroutines are called with MODE = -1 for their nominal parameter settings. Then the mode flag is incremented and a second call (MODE = 0) is made to the subroutines for their initialization. The mode switch is incremented again (MODE = 1) which corresponds to simulation. If linearization is not wanted during simulation SAC is set to 1, and call to subroutine LINK is bypassed. Otherwise LINK is called, and linearization is carried out at the beginning time point of the simulation. Then the number of integration steps NN between the simulation outputs is set, and the high-frequency computation loop (DO 400) is entered.

There are three exits from this loop. After each normal exit, that is after each ΔT simulation time interval, linear data is obtained by calling subroutine LINK, and a new simulation segment is started. The second type of exit occurs when the altitude variable h reaches the pull-up altitude h_p. The third type of exit occurs when total simulation time exceeds the specified maximum simulation time t_{max}.

In either of the first two types of exits, the program tests if weapon simulation is wanted. This is the normal case here ($SW \neq 0$). The program saves the values of the state x and its derivative $\dot{x}$, to initialize the weapon run. From this point on, similar information flow takes place for the weapon. At the end, control returns to the beginning of the program, and it expects to read a third run data. If it encounters a STOP card during input, it stops the program.

The subroutines used in the main program are listed in Table IX. For these subroutines which implement mathematical models, a reference is made to Volume I showing the page numbers of the pertinent analysis and modeling work.

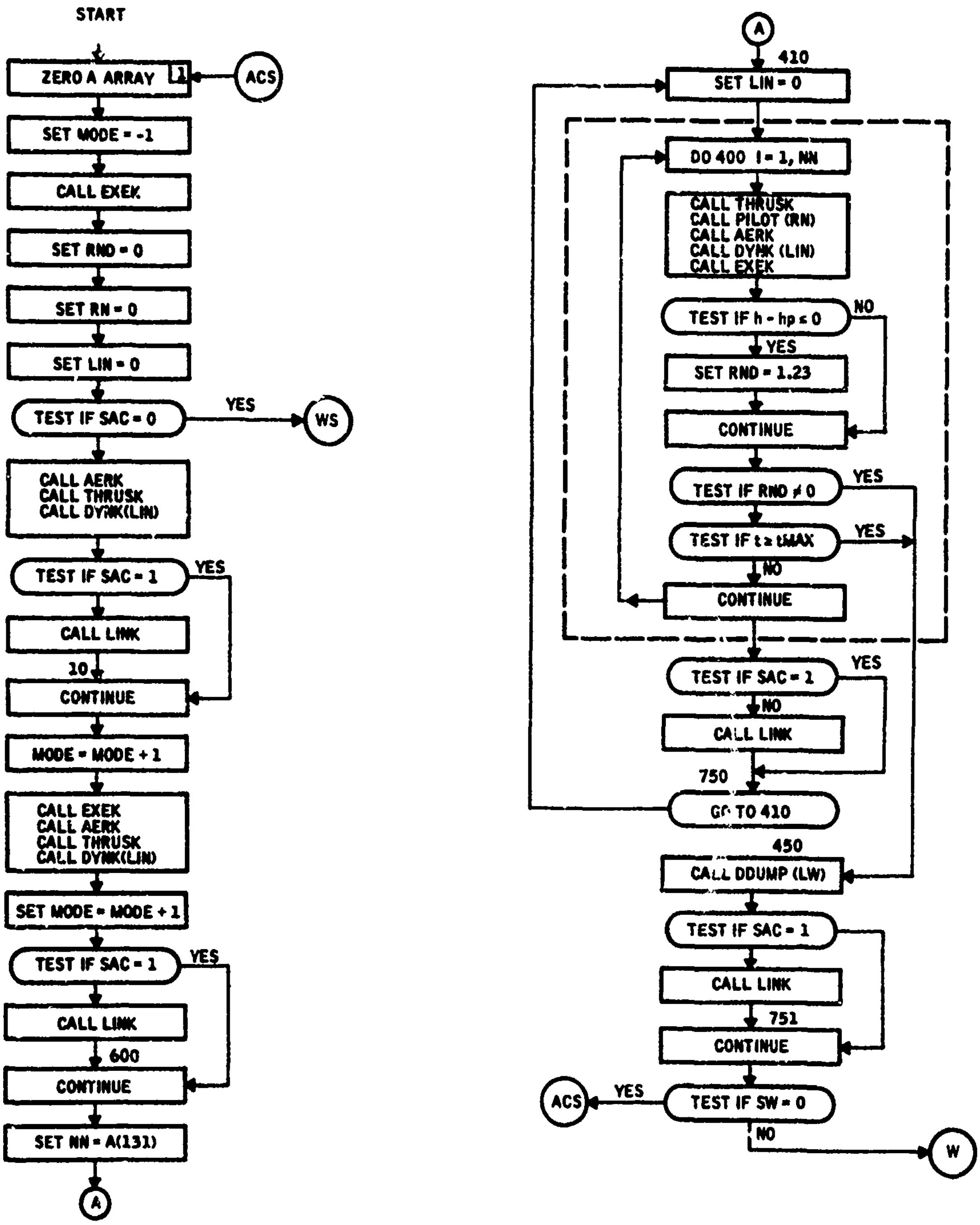


Figure 14. ADAP 1 Main Program Flow Diagram

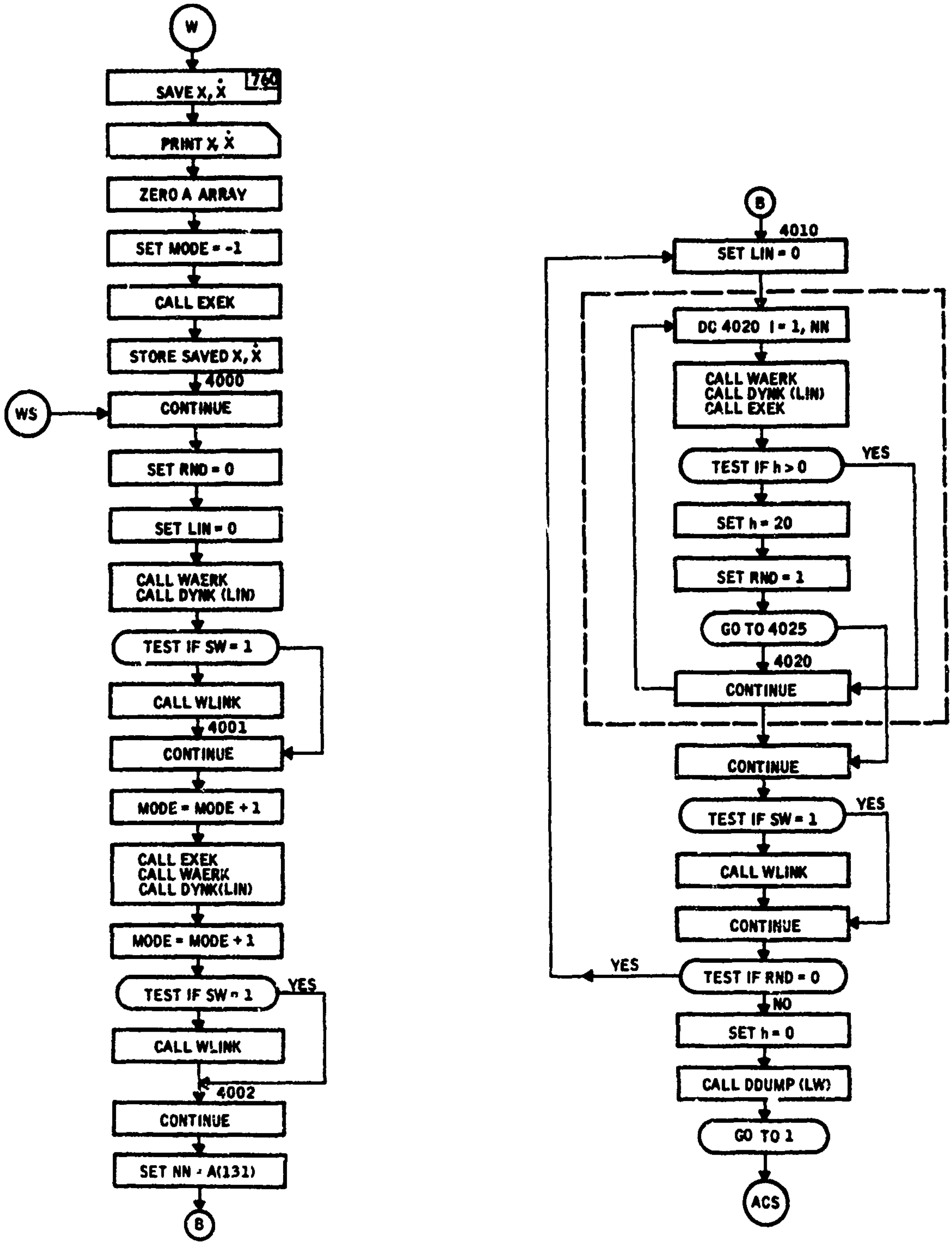


Figure 14. ADAP 1 Main Program Flow Diagram (concluded)

```
      PROGRAM ADAP(INPUT,OUTPUT,TAPE5=INPUT,TAPE9=OUTPUT,TAPE6,TAPE7)
C
C  ADAP  NON-LINEAR A/C AND WEAPON  MAIN PROGRAM
C
      COMMON/ADAP/MODE,A(1000)
      DIMENSION X(12),XDOT(12)
      EQUIVALENCE (SAC ,A(995)),(SW,A(996))
      LW=9
C
C ZERO A ARRAY
C
    1 DO 2 I=1,1000
    2 A(I)=0.
      MODE=-1
C
C CALL EXEK IN MODE=-1,THIS CALL TO EXEK READS IN     PARAMETER CHANGE
C CARDS AND TABLE DATA FOR  SUBROUTINE FLOOK
      CALL EXEK
      RND=0.
      RN=0.
      LIN=0
      IF(SAC.EQ.0.) GOTO 4000
      CALL AERK
      CALL THRUSK
      CALL DYNK(LIN)
      IF(SAC.EQ.1.) GOTO 10
      CALL LINK
   10 CONTINUE
      MODE=MODE+1
C
C CALL SUBROUTINES WITH MODE=0
C
      CALL EXEK
      CALL AERK
      CALL THRUSK
      CALL DYNK(LIN)
      MODF=MODE+1
      IF(SAC.EQ.1.) GOTO 600
      CALL LINK
  600 CONTINUE
      NN=A(131)
  410 LIN=0
C HIGH RATE LOOP FOR A/C NN ITEGRATION STEPS PER SEC.
      DO 400 I=1,NN
      CALL THRUSK
      CALL PILOT(RN)
      CALL AERK
      CALL DYNK(LIN)
      CALL EXEK
      IF(A(004)-A(274))702,702,703
  702 CONTINUE
      RND=1.23
  703 CONTINUE
      IF(RND.NE.0.) GOTO 450
```

Figure 15. ADAP 1 Main Program Input/Output Listing

```
      IF(A(001).GE.A(283)) GOTO 450
  400 CONTINUE
      IF(SAC.EQ.1.) GOTO 750
      CALL LINK
  750 GOTO 410
  450 CALL DDUMP(LW)
      IF(SAC.EQ.1.) GOTO 751
      CALL LINK
  751 CONTINUE
C
C  TEST FOR WEAPON SIMULATION FOLLOWING A/C SIMULATION
C
C  SAVE X AND XDOT OF A/C
      IF(SW.EQ.0.) GOTO 1
      DO 760 I=1,3
      I1=I+6
      I2=I+41
      I3=I+30
      I4=I+1
      I1D=I+16
      I2D=I+44
      I3D=I+38
      I4D=I+12
      X(I)=A(I1)
      X(I+3)=A(I2)
      X(I+6)=A(I3)
      X(I+9)=A(I4)
      XDOT(I)  =A(I1D)
      XDOT(I+3)=A(I2D)
      XDOT(I+6)=A(I3D)
      XDOT(I+9)=A(I4D)
  760 CONTINUE
C
C PRINT TERMINAL A/C STATE AND DERIVATIVE
C
      WRITE(LW,763)
  763 FORMAT(1H1/7X,21H A/C STATE AT RELEASE/)
      WRITE(LW,764)(X(I),I=1,12)
  764 FORMAT(/4H U =E15.8,4H V =E15.8,4H W =E15.8/4H P =E15.8,4H Q =E15.
     18,4H R =E15.8/7H THETA=E15.8,7H PHI  =E15.8,7H PSI  =E15.8/6H  XE
     2=E15.8,5H YE =E15.8,5H HE =E15.8)
      WRITE(LW,765)(XDOT(I),I=1,12)
  765 FORMAT(//5H UD =E15.8,5H VD =E15.8,5H WD =E15.8/5H PD =E15.8,5H QD
     1 =E15.8,5H RD =E15.8/8H THETAD=E15.8,8H PHID  =E15.8,8H PSID  =E15
     2.8/6H XED =E15.8,6H YED =E15.8,6H HED =E15.8)
      DO 761 I=1,1000
  761 A(I)=0.
      MODE=-1
C
C READ IN W  PON DATA
      CALL E. .
C
C STORE SAVED A/C STATE
C
```

Figure 15. ADAP 1 Main Program Input/Output Listing (continued)

```
      DO 762 I=1,3
      I1=I+6
      I2=I+41
      I3=I+30
      I4=I+1
      A(I1)=X(I)
      A(I2)=X(I+3)
      A(I3)=X(I+6)
      A(I4)=X(I+9)
  762 CONTINUE
 4000 CONTINUE
      RND=0.
      LIN=0
      CALL WAERK
      CALL DYNK(LIN)
      IF(SW.EQ.1.) GOTO 4001
      CALL WLINK
 4001 CONTINUE
      MODE=MODE+1
      CALL EXEK
      CALL WAERK
      CALL DYNK(LIN)
      MODE=MODE+1
      IF(SW.EQ.1.) GOTO 4002
      CALL WLINK
 4002 CONTINUE
      NN=A(131)
 4010 LIN=0
C
C HIGH RATE LOOP FOR WEAPON NN INTEGRATION STEPS PER SECOND
C
      DO 4020 I=1,NN
      CALL WAERK
      CALL DYNK(LIN)
      CALL EXEK
      IF(A(004).GT.0.) GOTO 4020
      A(004)=20.
      RND=1.
      GOTO 4025
 4020 CONTINUE
 4025 CONTINUE
      IF(SW.EQ.1.) GOTO 4031
      CALL WLINK
 4031 CONTINUE
      IF(RND.EQ.0.) GOTO 4010
      A(004)=0.
      CALL DDUMP(LW)
      GOTO 1
      END
```

Figure 15. ADAP 1 Main Program Input/Output Listing (concluded)

Table IX. ADAP 1 Subroutine Summary

Subroutine	Description	Flow Diagram (Figure No.)	Program Listing (Figure No.)	List of Symbols (Table No.)	Vol. I Reference
ADAP 1 (Main Program)	Nonlinear simulation and linearization of aircraft and weapon	14	15	---	Sections I-VII
DYNK	Aircraft and weapon dynamics for six degrees of freedom	17	18	X	pp. 21-41
AERK	Aerodynamic forces and moments for aircraft in body axes	19	20	XII	pp. 42-49
WAERK	Aerodynamic forces and moments for bomb	21	22	XIV	pp. 49-52
THRUSK	Thrust forces and moments for aircraft in body axes	23	24	XV	pp. 52-57
WINDK	Wind velocities for aircraft and weapon in body axes	25	26	XVI	pp. 58-71
SENK	Sensor kinematics of aircraft	27	28	---	pp. 72-87
PILOT	Trimming with autopilot	29	30	---	p. 124
NOMK*	Nominal parameters by algebraic trim	31	---	---	pp. 111-123
RELK*	Nominal weapon release	32	---	---	pp. 180-186
LINK	Linearization of aircraft dynamics for six degrees of freedom	33	34	XVII	pp. 88-104
SLINK	Linearization of aircraft sensor kinematics	35	36	---	p. 12-13
WLINK	Linearization of weapon dynamics for six degrees of freedom	---	37	---	pp. 88-110
EXEK	Executive	38	39	---	---
FLOOK	Table input look-up and interpolation	40	41	XVIII	---
PREAD	A-array parameter input	42	43	---	---
PRINT	A-array print	44	45	---	---
DDUMP	A-array data dump	46	47	---	---

*Not implemented in this study.

ADAP 1 SUBROUTINE STRUCTURE

Every subroutine used in ADAP 1 has the structure shown in Figure 16.

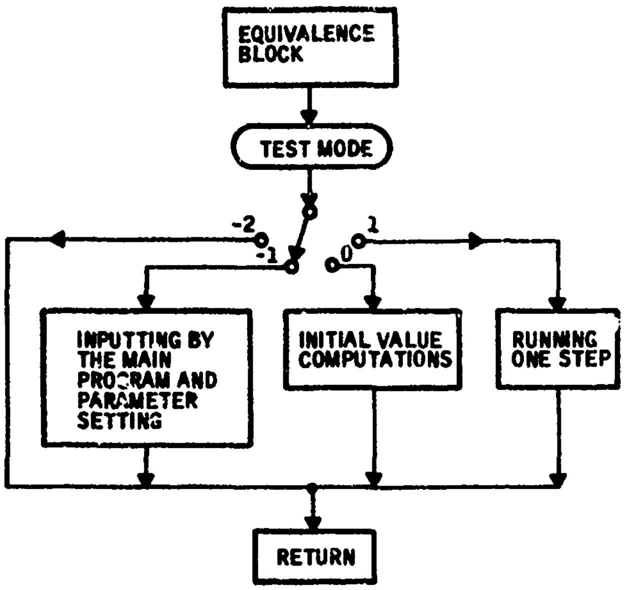


Figure 16. Subroutine Structure

The mode flag is initially set by the main program. The mode flag starts at -1 and is incremented in the main program by one on each pass through the subroutine until it reaches 1. It stays at 1 until the end conditions are reached, at which time it is set to -2.

ADAP 1 BASIC SUBROUTINES

Subroutine DYNK

Subroutine DYNK implements the model developed in Section III of Volume I. It combines the externally applied forces and moments with the aircraft kinematics and integrates the resulting differential equations of motion. The external forces and moments consist of components due to gravity, engine, steady-state and gusting wind, and aerodynamics. The kinematics include all cross products of inertia, which means the aircraft can be asymmetric about any axis. The subroutine assumes that aircraft body axes are used. This means that forces, moments and distance supplied as input are along aircraft body axes and likewise the outputs are with respect to body axes.

The subroutine DYNK flow diagram is shown in Figure 17 and the program listing in Figure 18. Symbols are listed and defined in Table X.

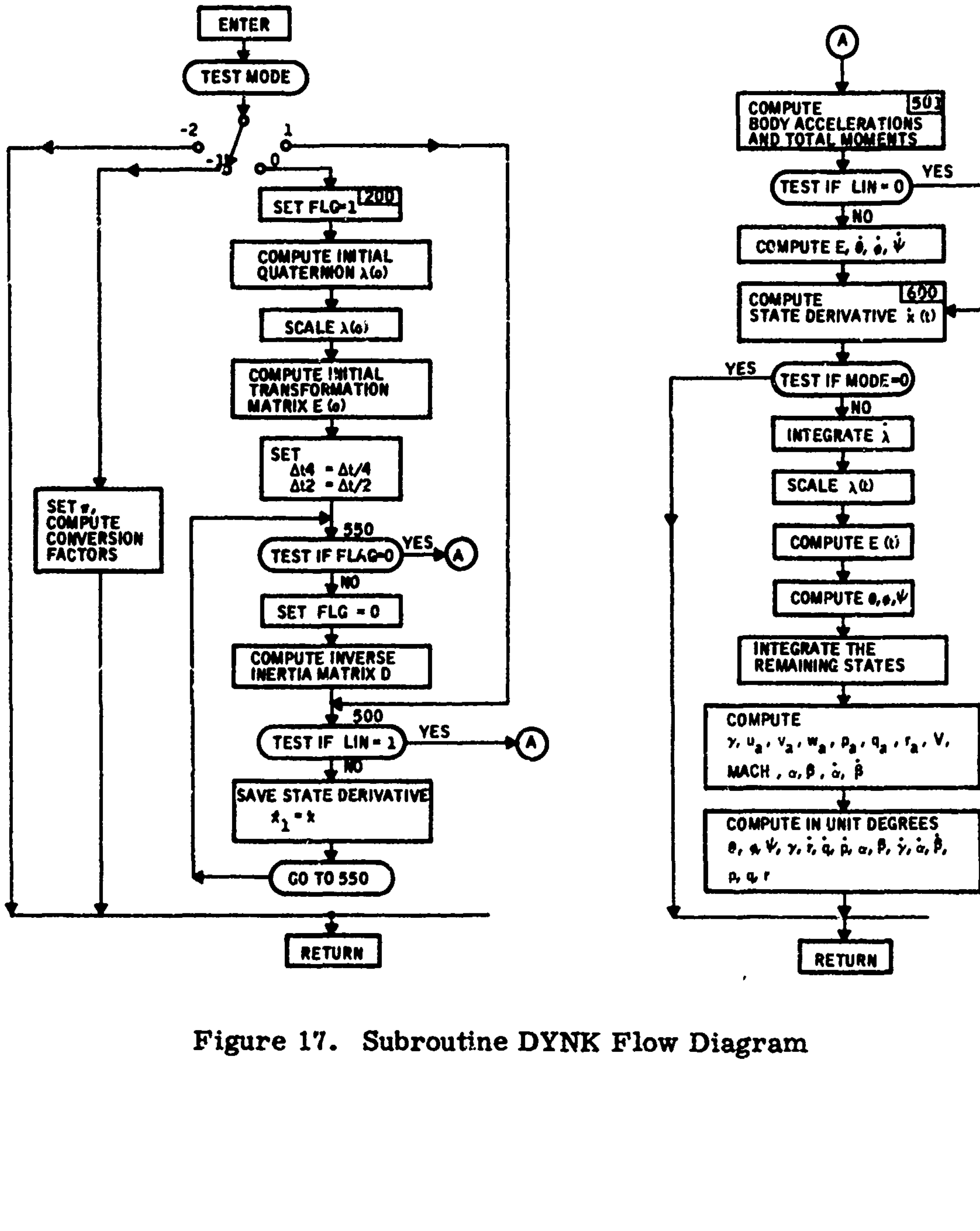


Figure 17. Subroutine DYNK Flow Diagram

```
      SUBROUTINE DYNK(LIN)
      COMMON/ADAP/MODE,A(1000)
C
C                         **** VARIABLE INPUTS ****
C
      EQUIVALENCE (SOS    ,A(006)),                 (FX     ,A(071)),
     1            (FY     ,A(072)),(FZ     ,A(073)),(L      ,A(077)),
     2            (M      ,A(078)),(N      ,A(079)),(IX     ,A(083)),
     3            (IY     ,A(084)),(IZ     ,A(085)),(IXY    ,A(086)),
     4            (IYZ    ,A(087)),(IXZ    ,A(088)),(DELT   ,A(089)),
     5            (G      ,A(090)),(MASS   ,A(091)),
     6                             (FLG    ,A(094)),(UG     ,A(101)),
     7            (VG     ,A(102)),(WG     ,A(103)),(RLT    ,A(600)),
     8            (RMT    ,A(601)),(RNT    ,A(602)),
     9            (XT     ,A(111)),(YT     ,A(113)),(ZT     ,A(112))
     A           ,(PG     ,A(104)),(QG     ,A(105)),(RG     ,A(106))
C
C                         **** VARIABLE OUTPUTS ****
C
      EQUIVALENCE (X      ,A(002)),(Y      ,A(003)),(H      ,A(004)),
     1            (MACH   ,A(005)),(U      ,A(007)),(V      ,A(008)),
     2            (W      ,A(009)),(UA     ,A(010)),(VA     ,A(011)),
     3            (WA     ,A(012)),(XDOT   ,A(013)),(YDOT   ,A(014)),
     4            (HDOT   ,A(015)),(VEL    ,A(016)),(UDOT   ,A(017)),
     5            (VDOT   ,A(018)),(WDOT   ,A(019)),(ACGX   ,A(020)),
     6            (ACGY   ,A(021)),(ACGZ   ,A(022)),(GAM    ,A(028)),
     7            (AL     ,A(029)),(BET    ,A(030)),(TH     ,A(031)),
     8            (PHI    ,A(032)),(PSI    ,A(033)),(GAMDOT,A(036)),
     9            (ALDOT ,A(037)),(BETDOT,A(038)),(P      ,A(042)),
     A            (Q      ,A(043)),(R      ,A(044)),(PDOT   ,A(045)),
     B            (QDOT   ,A(046)),(RDOT   ,A(047)),(STH    ,A(048)),
     C            (CTH    ,A(049)),(SPHI   ,A(050)),(CPHI   ,A(051)),
     D            (SPSI   ,A(052)),(CPSI   ,A(053)),(E11    ,A(054)),
     E            (E21    ,A(055)),(E31    ,A(056)),(E12    ,A(057)),
     F            (E22    ,A(058)),(E32    ,A(059)),(E13    ,A(060)),
     G            (E23    ,A(061)),(E33    ,A(062)),(CHI    ,A(066)),
     H            (THD    ,A(700)),(PHID   ,A(701)),(PSID   ,A(702)),
     I            (GAMD   ,A(703)),(RDOTD  ,A(704)),(QDOTD  ,A(705))
      EQUIVALENCE (ALD    ,A(706)),(BETD   ,A(707)),(GAMDTD,A(708)),
     1            (ALDOTD,A(709)),(BETDTD,A(710)),(PD     ,A(711)),
     2            (QD     ,A(712)),(RD     ,A(713)),(CHID   ,A(714))
     3           ,(PDOTD  ,A(715))
     4           ,(W1      A(161)),(W2     ,A(162)),(W3     ,A(163))
     5           ,(W4      A(164)),(W1DOT  ,A(165)),(W2DOT  ,A(166))
     6           ,(W3DOT  ,A(167)),(W4DOT  ,A(168)),(THDOT  ,A(039))
     7           ,(PHIDOT,A(040)),(PSIDOT,A(041)),(ELIN   ,A(259))
     8           ,(HDOT1  ,A(108)),(PA     ,A(080)),(QA     ,A(081))
     9           ,(RA     ,A(082)),(DELT4  ,A(265)),(DHDLT  ,A(266))
C
C
      REAL L,M,N,IX,IY,IZ,IXY,IYZ,IXZ,MASS,IMOM,MACH,MT,LT,NT
      IF (MODE) 100,200,500
  100 IF (MODE.LE.-2) RETURN
      PI=3.141592653589793
```

Figure 18. Subroutine DYNK Program Listing

```
      PIC=PI/180.0
      PICIN=1.0/PIC
      RETURN
  200 FLG    = 1.0
      GAMZ   = GAM
      IMOM   = 0.0
      SPHI=SIN(PHI)
      CPHI=COS(PHI)
      STH2   = SIN(0.5*TH)
      CTH2   = COS(0.5*TH)
      SPH2   = SIN(0.5*PHI)
      CPH2   = COS(0.5*PHI)
      SPS2   = SIN(0.5*PSI)
      CPS2   = COS(0.5*PSI)
      W1     = CPS2*CTH2*CPH2 + SPS2*STH2*SPH2
      W2     = CPS2*CTH2*SPH2 - SPS2*STH2*CPH2
      W3     = CPS2*STH2*CPH2 + SPS2*CTH2*SPH2
      W4     = -CPS2*STH2*SPH2 + SPS2*CTH2*CPH2
      W1W2   = W1*W2
      W1W3   = W1*W3
      W1W4   = W1*W4
      W2W3   = W2*W3
      W2W4   = W2*W4
      W3W4   = W3*W4
      W1SQ   = [illegible]1
      W2SQ   = [illegible]
      W3SQ   = W[illegible]W3
      W4SQ   = W4*W4
      FNORM  = 1.0/SQRT(W1SQ + W2SQ + W3SQ + W4SQ)
      W1     = W1*FNORM
      W2     = W2*FNORM
      W3     = W3*FNORM
      W4     = W4*FNORM
      E11    = (W1SQ + W2SQ - W3SQ - W4SQ)
      E12    = (W2W3 + W1W4)*2.0
      E13    = (W2W4 - W1W3)*2.0
      E21    = (W2W3 - W1W4)*2.0
      E22    = (W1SQ - W2SQ + W3SQ - W4SQ)
      E23    = (W3W4 + W1W2)*2.0
      E31    = (W2W4 + W1W3)*2.0
      E32    = (W3W4 - W1W2)*2.0
      E33    = (W1SQ - W2SQ - W3SQ + W4SQ)
      DELT4  = DELT/4.0
      DHDLT  = 0.5*DELT
  550 IF(FLG.EQ.0.0) GOTO 501
      FLG    = 0.0
      TMP4   = 1.0/(IX*IY)
      TMP5   = 1.0/(IY*IZ)
      TMP6   = 1.0/(IX*IZ)
      DK     = 1.0 - IXY*IXY*TMP4 - IYZ*IYZ*TMP5 - IXZ*IXZ*TMP6 -
     1         2.0*IXY*IYZ*IXZ*TMP4/IZ
      DAP1   = (1.0 - IYZ*IYZ*TMP5)/(IX*DK)
      DAP2   = (IXY + IYZ*IXZ/IZ)*TMP4/DK
      DAP3   = (IXZ + IXY*IYZ/IY)*TMP6/DK
```

Figure 18. Subroutine DYNK Program Listing (continued)

```
      DAQ1   = DAP2
      DAQ2   = (1.0 - IXZ*IXZ*TMP6)/(IY*DK)
      DAQ3   = (IYZ + IXZ*IXY/IX)*TMP5/DK
      DAR1   = DAP3
      DAR2   = DAQ3
      DAR3   = (1.0 - IXY*IXY*TMP4)/(IZ*DK)
  500 IF(LIN.EQ.1) GOTO 501
      PDOT1  = PDOT
      QDOT1  = QDOT
      RDOT1  = RDOT
      UDOT1  = UDOT
      VDOT1  = VDOT
      WDOT1  = WDOT
      W1DOT1 = W1DOT
      W2DOT1 = W2DOT
      W3DOT1 = W3DOT
      W4DOT1 = W4DOT
      XDOT1  = XDOT
      YDOT1  = YDOT
      HDOT1  = HDOT
      GOTO 550
  502 ELIN=0.
  501 ACGX   = (XT+FX)/MASS
      ACGY=(YT+FY)/MASS
      ACGZ=(ZT+FZ)/MASS
      LT=RLT+L
      MT=RMT+M
      NT=RNT+N
      IF(LIN.EQ.0) GOTO 600
      STH=SIN(TH)
      CTH=COS(TH)
      SPHI=SIN(PHI)
      CPHI=COS(PHI)
      SPSI=SIN(PSI)
      CPSI=COS(PSI)
      E11=CTH*CPSI
      E12=CTH*SPSI
      E13=-STH
      E21=-CPHI*SPSI+SPHI*STH*CPSI
      E22= CPHI*CPSI+SPHI*STH*SPSI
      E23= SPHI*CTH
      E31= SPHI*SPSI+CPHI*STH*CPSI
      E32=-SPHI*CPSI+CPHI*STH*SPSI
      E33= CPHI*CTH
      SECTH  = 1./CTH
      TANTH  = STH/CTH
      THDOT  = CPHI*Q-SPHI*R
      PHIDOT = P+SPHI*TANTH*Q+CPHI*TANTH*R
      PSIDOT = SPHI*SECTH*Q+CPHI*SECTH*R
  600 DA1    =LT + Q*R*(IY - IZ) - IYZ*(R*R - Q*Q) + P*(IXZ*Q - IXY*R)
      DA2    = MT + R*(P*(IZ - IX) - IMOM) - IXZ*(P*P - R*R) +
     1         Q*(IXY*R - IYZ*P)
      DA3    =NT + Q*(P*(IX - IY) + IMOM) - IXY*(Q*Q - P*P) +
     1         R*(IYZ*P - IXZ*Q)
```

Figure 18. Subroutine DYNK Program Listing (continued)

```
C
C               COMPUTATION OF BODY AXES VELOCITIES AND ANGULAR ACCELERATIONS
C
      UDOT   = ACGX + G*E13 - W*Q + V*R
      VDOT   = ACGY + G*E23 - U*R + W*P
      WDOT   = ACGZ + G*E33 - V*P + U*Q
      PDOT   = DA1*DAP1 + DA2*DAP2 + DA3*DAP3
      QDOT   = DA1*DAQ1 + DA2*DAQ2 + DA3*DAQ3
      RDOT   = DA1*DAR1 + DA2*DAR2 + DA3*DAR3
      A(661)=U/1.69
C
C               COMPUTE A/C VELOCITIES W.R.T. EARTH IN EARTH COORDINATES
C
      XDOT   = E11*U+E21*V+E31*W
      YDOT   = E12*U+E22*V+E32*W
      HDOT   =-E13*U-E23*V-E33*W
      IF(LIN.EQ.1) GOTO 2000
      W1DOT  = -P*W2 - Q*W3 - R*W4
      W2DOT  =  P*W1 + R*W3 - Q*W4
      W3DOT  =  Q*W1 - R*W2 + P*W4
      W4DOT  =  R*W1 + Q*W2 - P*W3
      IF(MODE.EQ.0) RETURN
C
C               COMPUTE EULER SYMMETRICAL PARAMETERS
C
      W1     = W1 + DELT4*(3.0*W1DOT - W1DOT1)
      W2     = W2 + DELT4*(3.0*W2DOT - W2DOT1)
      W3     = W3 + DELT4*(3.0*W3DOT - W3DOT1)
      W4     = W4 + DELT4*(3.0*W4DOT - W4DOT1)
C
C               COMPUTE EARTH TO BODY ROTATION MATRIX (E)
C
 1000 W1W2   = W1*W2
      W1W3   = W1*W3
      W1W4   = W1*W4
      W2W3   = W2*W3
      W2W4   = W2*W4
      W3W4   = W3*W4
      W1SQ   = W1*W1
      W2SQ   = W2*W2
      W3SQ   = W3*W3
      W4SQ   = W4*W4
      FNORM  = 1.0/SQRT(W1SQ + W2SQ + W3SQ + W4SQ)
      W1     = W1*FNORM
      W2     = W2*FNORM
      W3     = W3*FNORM
      W4     = W4*FNORM
      E11    = (W1SQ + W2SQ - W3SQ - W4SQ)
      E12    = (W2W3 + W1W4)*2.0
      E13    = (W2W4 - W1W3)*2.0
      E21    = (W2W3 - W1W4)*2.0
      E22    = (W1SQ - W2SQ + W3SQ - W4SQ)
      E23    = (W3W4 + W1W2)*2.0
      E31    = (W2W4 + W1W3)*2.0
```

Figure 18. Subroutine DYNK Program Listing (continued)

```
      E32     = (W3W4 - W1W2)*2.0
      E33     = (W1SQ - W2SQ - W3SQ + W4SQ)
      IF(ABS(E13).GT.1.) GOTO 2015
      ARG2= SQRT(1.-E13*E13)
      TH=-ATAN2(E13,ARG2)
      GOTO 2016
 2015 E13=ABS(E13)/E13
 2016 CONTINUE
      PHI     = ATAN2(E23,E33)
      IF (PHI.GT.3.141593) PHI = PHI - 6.283186
      SPHI    = SIN(PHI)
      CPHI    = COS(PHI)
      PSI     = ATAN2(E12,E11)
      IF (PSI.GT.3.141593) PSI = PSI - 6.283186
C
C                 INTEGRATE BODY VELOCITIES AND ANGULAR ACCELERATIONS
C
      U       = U + DHDLT*(3.0*UDOT - UDOT1)
      V       = V + DHDLT*(3.0*VDOT - VDOT1)
      W       = W + DHDLT*(3.0*WDOT - WDOT1)
      P       = P + DHDLT*(3.0*PDOT - PDOT1)
      Q       = Q + DHDLT*(3.0*QDOT - QDOT1)
      R       = R + DHDLT*(3.0*RDOT - RDOT1)
C
C                 INTEGRATE EARTH VELOCITIES
C
      X       = X + DHDLT*(3.0*XDOT - XDOT1)
      Y       = Y + DHDLT*(3.0*YDOT - YDOT1)
      H       = H + DHDLT*(3.0*HDOT - HDOT1)
      CHI     = ATAN2(YDOT,XDOT)
      IF (CHI.GT.3.141593) CHI = CHI - 6.283186
      GAM     = ATAN (HDOT/SQRT(XDOT*XDOT + YDOT*YDOT))
C
C                 INCORPORATE WIND COMPONENTS
C
      UA      = U - UG
      VA      = V - VG
      WA      = W - WG
      PA=P-PG
      QA=Q-QG
      RA=R-RG
      TMP1    = UA*UA + WA*WA
      TMP2    = SQRT(TMP1)
      TMP3    = TMP1 + VA*VA
      VEL     = SQRT(TMP3)
      MACH    = VEL/SOS
      AL      = ATAN(WA/UA)
      BET     = ATAN(VA/TMP2)
      ALDOT   = (UA*WDOT - WA*UDOT)/TMP1
      BETDOT = (TMP1*VDOT - VA*(UA*UDOT + WA*WDOT))/(TMP2*TMP3)
      THD = TH*PICIN
      PHID = PHI*PICIN
      PSID = PSI*PICIN
      GAMD = GAM*PICIN
```

Figure 18. Subroutine DYNK Program Listing (continued)

```
      RDOTD = RDOT*PICIN
      QDOTD = QDOT*PICIN
      PDOTD  = PDOT*PICIN
      ALD = AL*PICIN
      BETD = BET*PICIN
      GAMDTD = GAMDOT*PICIN
      ALDOTD = ALDOT*PICIN
      BETDTD = BETDOT*PICIN
      PD = P*PICIN
      QD = Q*PICIN
      RD = R*PICIN
      CHID = CHI*PICIN
      RETURN
 2000 CONTINUE
      RETURN
      END
```

Figure 18. Subroutine DYNK Program Listing (concluded)

Table X. List of Symbols for Subroutine DYNK

Quantity	Mnemonic	A-Array Index	Initial Value	Units	Input	Output	Description
a_x	ACGX	20		ft/sec^2		X	Acceleration of the body cg due to all forces, except gravity, along the body axes x, y, z
a_y	ACGY	21		ft/sec^2		X	
a_z	ACGZ	22		ft/sec^2		X	
α_a	AL	29		rad		X	Angle of attack wrt instantaneous air mass
α°_a	ALD	706		deg		X	Angle of attack wrt instantaneous air mass
$\dot{\alpha}_a$	ALDOT	37		rad/sec		X	Scalar time derivative of α_a
$\dot{\alpha}^\circ_a$	ALDOTD	709		deg/sec		X	Scalar time derivative of α°_a
β_a	BET	30		rad		X	Sideslip angle wrt instantaneous air mass
β°_a	BETD	707		deg		X	Sideslip angle wrt instantaneous air mass
$\dot{\beta}_a$	BETDOT	38		rad/sec		X	Scalar time derivative of β_a
$\dot{\beta}^\circ_a$	BETDTD	710		deg/sec		X	Scalar time derivative of β°_a
$\cos\phi$	CPHI	51		N/A		X	Cosines of the Euler angles ϕ, ψ and θ
$\cos\phi/2$	CPH2	---		N/A			
$\cos\Psi$	CPS1	53		N/A		X	

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
$\cos \Psi/2$	CPS2				N/A	
$\cos \theta$	CTH	49			X	
$\cos \theta/2$	CTH2					
d_{11}	DAP1					Elements of inverse inertia data matrix
d_{12}	DAP2					
d_{13}	DAP3					
d_{21}	DAQ1					
d_{22}	DAQ2					
d_{23}	DAQ3					
d_{31}	DAR1					
d_{32}	DAR2					
d_{33}	DAR3					
f1	DA1					Intermediate variables defined on p. 36 of Vol. I
f2	DA2					
f3	DA3					
Δt	DELT	89	sec	X		Integration step size
$\Delta t/4$	DELT4		sec			
$\Delta t/2$	DHDLT		sec			

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
ΔX_T	DXT	92	ft	X		Moment arm for z component of engine thrust, i. e., distance from cg
ΔZ_T	DZT	93	ft	X		Moment arm for x component of thrust, i. e., distance from cg
e_{11}	E11	54			X	Elements of direction cosine matrix transforming a vector from earth axes to body axes
e_{12}	E12	57			X	
e_{13}	E13	60			X	
e_{21}	E21	55			X	
e_{22}	E22	58			X	
e_{23}	E23	61			X	
e_{31}	E31	56			X	
e_{32}	E32	59			X	
e_{33}	E33	62			X	
flag	FLG	94		X		If body moments of inertia change during running, this flag should be set to 1

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Initial Value	Units	Input	Output	Description
F_x	FX	71		lb	X		Aerodynamic forces along the x, u, z body axes
F_y	FY	72		lb	X		
F_z	FZ	73		lb	X		
g	G	90		ft/sec^2			Gravitational acceleration
γ	GAM	28		rad		X	Flight path angle wrt earth
$\gamma°$	GAMD	703		deg		X	Flight path angle wrt earth
H	H	4		ft		X	Body altitude (along $-z_e$ earth axis)
$\dot{H}$	HDOT	15		ft/sec		X	Component of body velocity wrt earth along earth axis $-z_e$ (altitude rate)
$\dot{H}_{-1}$	HDOT1			ft/sec			Past value of $\dot{H}$
I_{xx}	IX	83		$slug\text{-}ft^2$	X		Moments of inertia of the body about body axes located at the cg
I_{xy}	IXY	86			X		
I_{xz}	IXZ	88			X		
I_{yy}	IY	84			X		
I_{yz}	IYZ	87			X		
I_{zz}	IZ	85			X		

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Initial Value	Units	Input	Output	Description
L	L	77		ft-lb	X		Aerodynamic moment about the body axis x
M	M	78		ft-lb	X		Aerodynamic moment about the body axis y
Ma	MACH	5				X	Mach number wrt instantaneous wind
m	MASS	91		slugs	X		Total mass of body
Mode	MODE						Flag for running modes (see subroutine description)
N	N	79		ft-lb	X		Aerodynamic moment about the body axis z
p	P	42		rad/sec		X	A component of body angular rate about body axis x
p°	PD	711		deg/sec		X	
$\dot{p}$	PDOT	45		rad/sec^2		X	A component of body angular acceleration about body axis x
$\dot{p}^{\circ}$	PDOTD	715		deg/sec^2		X	
$\dot{p}_{-1}$	PDOT1	---					Past value of $\dot{p}$
ϕ	PHI	32		rad		X	Euler roll angle
ϕ°	PHID	701		deg		X	Euler roll angle

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
π	PI		rad			
$\pi/180$	PIC		rad/deg			
$(\pi/180)^{-1}$	PICIN		deg/rad			
ψ	PSI	33	rad		X	Euler yaw angle
ψ°	PSID	702	deg		X	Euler yaw angle
q	Q	43	rad/sec		X	A component of body angular rate about body axis y
	QBAR	70		X		Dynamic pressure
q°	QD	712	deg/sec		X	Component of body angular rate about body axis y
$\dot{q}$	QDOT	46	rad/sec^2		X	A component of body angular acceleration about body axis y
$\dot{q}^\circ$	QDOTD	705	deg/sec^2		X	↓
$\dot{q}_{-1}$	QDOT1		rad/sec^2			Past value of $\dot{q}$
r	R	44	rad/sec		X	A component of body angular rate about body axis z
r°	RD	713	deg/sec		X	↓
$\dot{r}$	RDOT	47	rad/sec^2		X	A component of body angular rate about body axis z
$\dot{r}^\circ$	RDOTD	704	deg/sec^2		X	↓
$\dot{r}_{-1}$	RDOT1					Past value of $\dot{r}$

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
L_T	RLT	600	ft-lb	X		Total thrust moments about x, y, z axes
M_T	RMT	601	ft-lb	X		
N_T	RNT	602	ft-lb	X		
a	SOS	6, 854	ft/sec	X		Speed of sound
$\sin \phi$	SPHI	50			X	Sines of the Euler angles ϕ, ψ, and θ
$\sin \phi/2$	SPH2					
$\sin \Psi$	SPSI	52			X	
$\sin \Psi/2$	SPS2					
$\sin \theta$	STH	48			X	
$\sin \theta/2$	STH2					
θ	TH	31	rad		X	Euler pitch angle
θ_d	THD	700	deg		X	Euler pitch angle
u	U	7	ft/sec		X	Velocity component of cg along x axis wrt earth
u_a	UA	10	ft/sec		X	Velocity component of cg along x axis wrt air mass
$\dot{u}$	UDOT	17	ft/sec^2		X	Time derivative of u
$\dot{u}_{-1}$	UDOT1		ft/sec^2			Past value of $\dot{u}$

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
u_g	UG	101	ft/sec	X		Wind gust velocity along x axis
v	v	8	ft/sec		X	Velocity component of cg along y axis wrt earth
v_a	VA	11	ft/sec		X	Velocity component of cg along y axis wrt air mass
$\dot{v}$	VDOT	18	ft/sec^2		X	Time derivative of v
$\dot{v}_{-1}$	VDOT1		ft/sec^2			Past value of $\dot{v}$
	VEL	16	ft/sec		X	Magnitude of velocity vector
v_g	VG	102	ft/sec	X		Wind gust velocity along y axis
w	W	9	ft/sec		X	Velocity component of cg along z axis wrt earth
w_a	WA	12	ft/sec		X	Velocity component of cg along z axis wrt air mass
$\dot{w}$	WDOT	19	ft/sec^2		X	Time derivative of w
$\dot{w}_{-1}$	WDOT1		ft/sec^2			Past value of $\dot{w}$
w_g	WG	103	ft/sec	X		Wind gust velocity along z axis
λ_o	W1				X	First component of angular coordinate (quaternion)
$\dot{\lambda}_o$	WIDOT				X	Time derivative of λ_o
$\dot{\lambda}_{o-1}$	WIDOTI				X	Past value of $\dot{\lambda}_o$

Table X. List of Symbols for Subroutine DYNK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
λ_o^2	W1SQ				X	Square of λ_o
$\lambda_o\lambda_1$	W1W2					Intermediate variables
$\lambda_o\lambda_2$	W1W3					
$\lambda_o\lambda_3$	W1W4					
λ_1	W2					Second component of quaternion
$\dot{\lambda}_1$	W2DOT					Time derivative of λ_1
$(\dot{\lambda}_1)_{-1}$	W2DOT1					Past value of $\dot{\lambda}_1$
λ_1^2	W2SQ					Square of λ_1
$\lambda_1\lambda_2$	W2W3					Intermediate variables
$\lambda_1\lambda_3$	W2W4					
λ_2	W3					Third component of quaternion
$\dot{\lambda}_2$	W3DOT					Time derivative of λ_2
$(\dot{\lambda}_2)_{-1}$	W3DOT1					Past value of $\dot{\lambda}_2$
λ_2^2	W3SQ					Square of λ_2
$\lambda_2\lambda_3$	W3W4					Intermediate variable

Table X. List of Symbols for Subroutine DYNK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
λ_3	W4				X	Fourth component of quaternion
$\dot{\lambda}_3$	W4DOT					Time derivative of λ_3
$(\dot{\lambda}_3)_{-1}$	W4DOT1					Past value of $\dot{\lambda}_3$
λ_3^2	W4SQ					Square of λ_3
x_e	X	2	ft		X	Body position along earth axis x_e
$\dot{x}_e$	XDOT	13	ft/sec		X	Velocity component along x_e
$\dot{x}_{e_{-1}}$	XDOTI		ft/sec			Past value of $\dot{x}_e$
T_x	XT	111	lb	X		Component of engine thrust along body axis x
y_e	Y	3	ft		X	Body position along earth axis y
$\dot{y}_e$	YDOT	14	ft/sec		X	Component of body velocity wrt earth along earth axis y_e
$\dot{y}_{e_{-1}}$	YDOT1		ft/sec			Past value of $\dot{y}_e$
T_y	YT	113	lb	X		Component of engine thrust along body axis y
T_z	ZT	112	lb	X		Component of engine thrust along body axis z

Subroutine AERK

Subroutine AERK implements the model developed in Section IV of Volume I. It is used to generate aircraft nonlinear aerodynamics, i.e., aerodynamics which are a function of empirical curves. A general-purpose subroutine to perform this function is impossible because there is no standard form for specifying nonlinear aerodynamics. However, this problem is reduced to its simplest form in ADAPS. The user just encodes his particular form of aerodynamic equations into subroutine AERK and sets up his aerodynamic data in function look-up form (Appendix I).

Aircraft coefficients and their look-up representations are listed in Table XI. The subroutine AERK flow diagram is shown in Figure 19 and the program listing in Figure 20. Symbols are defined in Table XII.

Subroutine WAERK

Subroutine WAERK implements the model developed in Section IV of Volume I. It is used to generate weapon nonlinear aerodynamics. In general, the user must encode his particular form of weapon aerodynamic equations into subroutine WAERK and must set up his weapon aerodynamic data in function look-up form (Appendix I).

Bomb aerodynamic coefficients and their look-up representations are listed in Table XIII. The subroutine WAERK flow diagram is shown in Figure 21 and the program listing in Figure 22. Symbols are listed in Table XIV.

Subroutine THRUSK

Subroutine THRUSK implements the model developed in Section IV of Volume I. It generates total forces and moments along aircraft body axes produced by the thrusters on the aircraft. In addition to two main jet engines, three thrust points are provided in the subroutine for simulating the vernier thrusting in high-performance aircraft.

The subroutine THRUSK flow diagram is shown in Figure 23 and the program listing in Figure 24. Symbols are listed in Table XV.

Table XI. Representation of Aircraft Coefficients in Stability Axes

Coefficient	Look-Up Representation	Mnemonic	Units
$C_L(M_a, h, \alpha^o{}_w)$	F1(1, 2, 3)	CL	
$C_{z_q}(h, M_a)$	F2(2, 1)	CZQ	per radian
$C_{z_{\dot{\alpha}}}(h, M_a)$	F3(2, 1)	CZALDT	per radian
$C_{L_{\delta_s}}(h, \alpha^o{}_w, M_a)$	F4(2, 3, 1)	CLDS	per degree
$C_{L_{\delta_{sp}}}(h, M_a)$	F5(2, 1)	CLDSP	per degree
$C_{L_{\delta_a}}(h, M_a)$	F6(2, 1)	CLDA	per degree
$C_{L_{\delta_{sB}}}(\alpha^o{}_w, M_a)$	F7(3, 1)	CLDSB	per degree
$C_{L_{\delta_{LG}}}(\alpha^o{}_w)$	F8(3)	CLDLG	
$C_D(Pow, M_a C_L)$	F9(4, 1, 5)	CD	
$C_D(\delta^o{}_{sB}, C_L, M_a)$	F10(6, 5, 1)	CDDSB	
$C_{D_{\delta_{LG}}}(CL)$	F11(5)	CDDLG	
$C_{mca}(M_a, h, \alpha^o{}_w)$	F12(1, 2, 3)	CMCA	
$C_{m_q}(h, M_a)$	F13(2, 1)	CMQ	per radian
$C_{m_{\dot{\alpha}}}(h, M_a)$	F14(2, 1)	CMALDT	per radian

Table XI. Representation of Aircraft Coefficients in Stability Axes (continued)

Coefficient	Look-Up Representation	Mnemonic	Units
$C_{m_{\delta_s}}(h, \alpha^o{}_w, M_a)$	F15(2, 3, 1)	CMDS	per degree
$C_{m_{\delta_{sp}}}(\alpha^o{}_w, h, M_a)$	F16(3, 2, 1)	CMDSP	per degree
$C_{m_{\delta_a}}(\alpha^o{}_w, h, M_a)$	F17(3, 2, 1)	CMDA	per degree
$C_{m_{\delta_{sB}}}(M_a, \alpha^o{}_w)$	F18(3, 1)	CMDSB	per degree
$C_{m_{\delta_{LG}}}(\alpha^o{}_w)$	F19(3)	CMDLG	
$C_{y_\beta}(\alpha^o{}_w, h, M_a)$	F20(3, 2, 1)	CYBET	per degree
$C_{y_r}(\alpha^o{}_w, h, M_a)$	F21(3, 2, 1)	CYR	per radian
$C_{y_p}(h, \alpha^o{}_w, M_a)$	F22(2, 3, 1)	CYP	per radian
$C_{y_{\delta_{sp}}}(M_a)$	F23(1)	CYDSP	per degree
$C_{y_{\delta_a}}(M_a)$	F24(1)	CYDA	per degree
$C_{y_{\delta_r}}(h, M_a)$	F25(2, 1)	CYDR	per degree
$C_{n_\beta}(\alpha^o{}_w, h, M_a)$	F26(3, 2, 1)	CNBET	per degree
$C_{n_r}(M_a, h, \alpha^o{}_w)$	F27(3, 2, 1)	CNR	per radian

Table XI. Representation of Aircraft Coefficients in Stability Axes (continued)

Coefficient	Look-Up Representation	Mnemonic	Units
$C_{n_p}(h, \alpha^o_w, M_a)$	F28(2, 3, 1)	CNP	per radian
$C_{n_{\delta_{sp}}}(\alpha^o_w, M_a)$	F29(3, 1)	CNDSP	per degree
$C_{n_{\delta_a}}(\alpha^o_w, h, M_a)$	F30(3, 2, 1)	CNDA	per degree
$C_{n_{\delta_r}}(h, M_a)$	F31(2, 1)	CNDR	per radian
$C_{1_\beta}(\alpha^o_w, h, M_a)$	F32(3, 2, 1)	CLLBET	per degree
$C_{1_r}(\alpha^o_w, h, M_a)$	F33(3, 2, 1)	CLLR	per radian
$C_{1_p}(\alpha^o_w, h, M_a)$	F34(3, 2, 1)	CLLP	per radian
$C_{1_{\delta_{sp}}}(\alpha^o_w, h, M_a)$	F35(3, 2, 1)	CLLDSP	per degree
$C_{1_{\delta_a}}(\alpha^o_w, h, M_a)$	F36(3, 2, 1)	CLLDA	per degree
$C_{1_{\delta_r}}(\alpha^o_w, h, M_a)$	F37(3, 2, 1)	CLLDR	per degree
$X_{c.g.}(Y_{trm}, wt)$	F45(10, 15)	XCG	inches
$Z_{c.g.}(Y_{trm}, wt)$	F46(10, 15)	ZCG	inches
$I_x(Y_{trm}, wt)$	F47(10, 15)	IX	slug/ft^2
$I_y(Y_{trm}, wt)$	F48(10, 15)	IY	slug/ft^2

Table XI. Representation of Aircraft Coefficients in Stability Axes (concluded)

Coefficient	Look-Up Representation	Mnemonic	Units
$I_z(Y_{trm}, W_T)$	F49(10, 15)	IZ	$slug/ft^2$
$I_{xz}(Y_{trm}, W_T)$	F50(10, 15)	IXZ	$slug/ft^2$
$\rho(h)$	F53(2)	RHO	lbs/ft^3
a(h)	F54(2)	SOS	ft/sec

Argument	Look-Up Representation	Mnemonic	Units
M	V(1)	Mach	
h	V(2)	H	feet
α^o_w	V(3)	AL+1°	degree
Power	V(4)	POWER	per unit
C_L	V(5)	CL	
δ_{sb}	V(6)	YDSB	degree
δ_s	V(7)	YDS	degree
wt	V(15)	WT	pounds

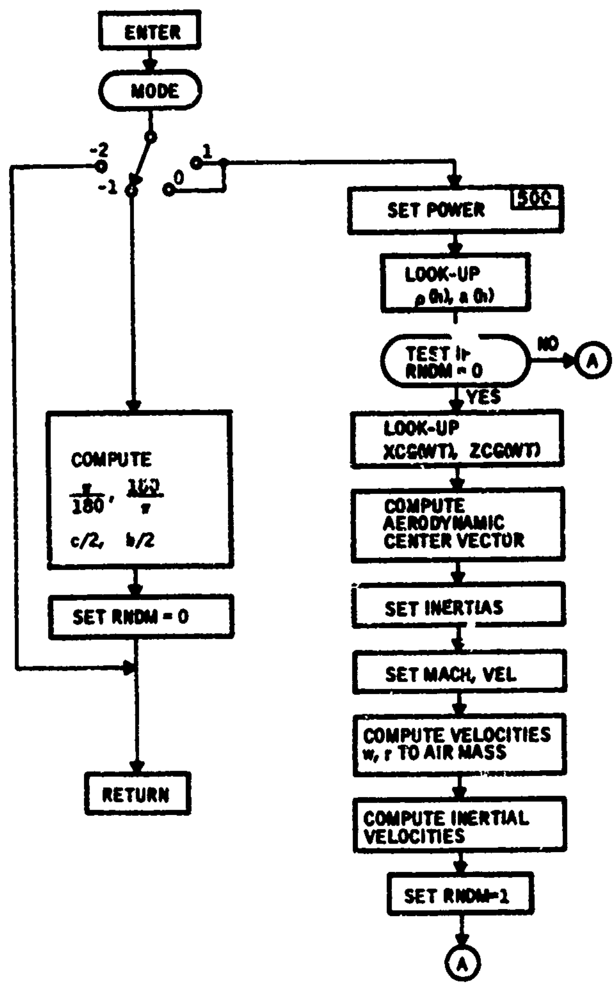


Figure 19. Subroutine AERK Flow Diagram

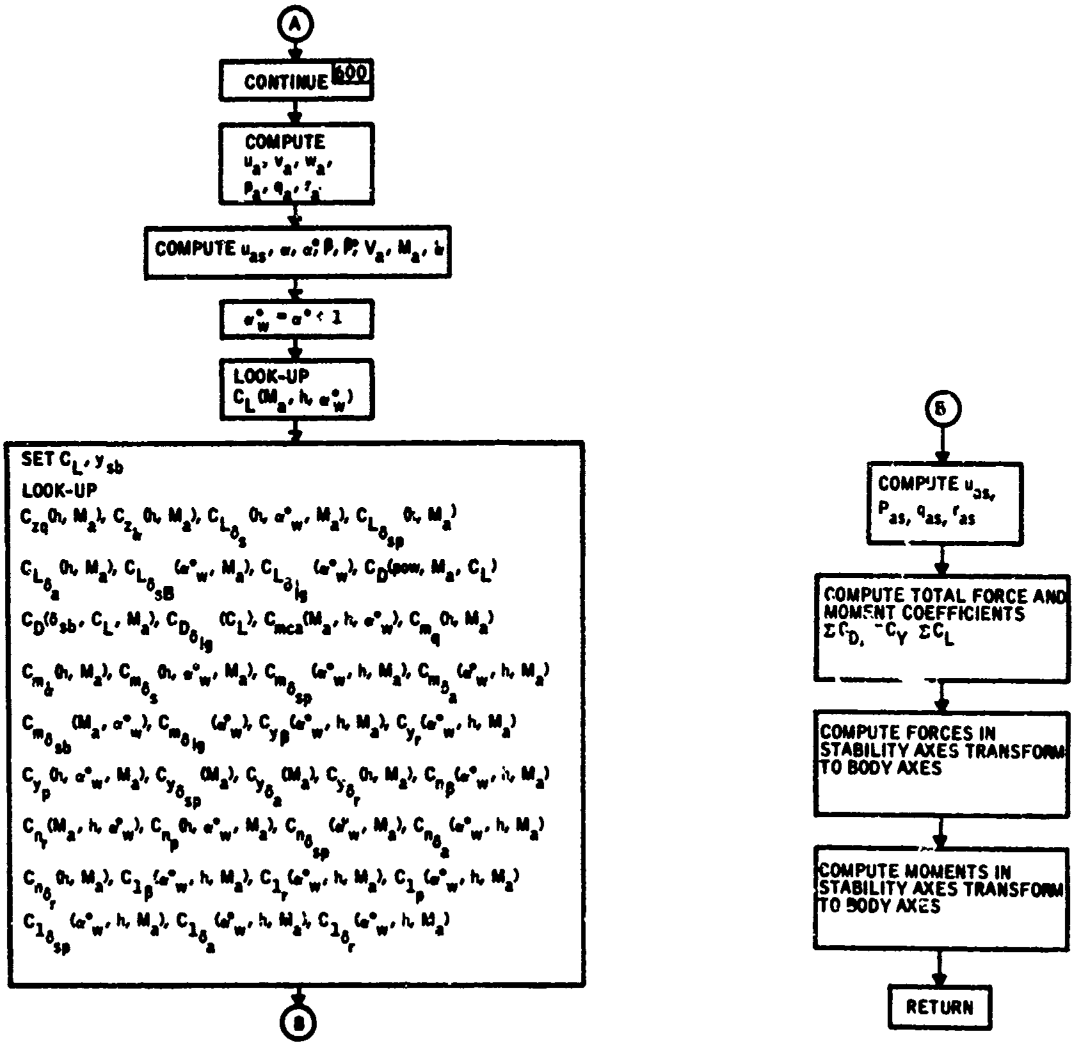


Figure 19. Subroutine AERK Flow Diagram (concluded)

```
      SUBROUTINE AERK
      DIMENSION XT(2)
      COMMON/ADAP/MODE,A(1000)
      DIMENSION V(20),F(80)
C                         **** PARAMETER INPUTS ****
C
C
      EQUIVALENCE (SOS    ,A(006)),(G      ,A(090)),(MASS   ,A(091)),
     1            (RHO    ,A(114)),(B      ,A(115)),(C      ,A(116)),
     2            (S      ,A(117))                 ,(WT     ,A(716))
     3           ,(XCA    ,A(579)),(YCA    ,A(580)),(ZCA    ,A(581))
     4                            ,(YTRM   ,A(651))
C
C                         **** VARIABLE  INPUTS ****
C
      EQUIVALENCE (H      ,A(004)),(MACH   ,A(005)),(VEL    ,A(016)),
     1            (AL     ,A(029)),(BET    ,A(030)),(ALDT   ,A(037)),
     2                             (PA     ,A(080)),(QA     ,A(081)),
     3            (RA     ,A(082)),(YDS    ,A(121)),(YDA    ,A(123)),
     4            (YDR    ,A(125)),(YSP    ,A(645)),(YSB    ,A(646)),
     5                             (YLG    ,A(648)),
     6                             (XT(1)  ,A(559)),(ALD    ,A(706)),
     7            (BETD   ,A(707)),(UG     ,A(101)),(WG     ,A(103))
     8           ,(VG     ,A(102)),(PG     ,A(104)),(QG     ,A(105)),
     9            (RG     ,A(106)),(U      ,A(007)),(VSIDE  ,A(008)),
     A            (W      ,A(009)),(P      ,A(042)),(Q      ,A(043)),
     B            (R      ,A(044)),(UDOT   ,A(017)),
     C            (WDOT   ,A(019))
C
C                         **** VARIABLE OUTPUTS ****
C
      EQUIVALENCE (UA     ,A(010)),(WA     ,A(012)),
     1            (QBAR   ,A(070)),(FX     ,A(071)),(FY     ,A(072)),
     2            (FZ     ,A(073)),(L      ,A(077)),(M      ,A(078)),
     3            (N      ,A(079)),(IX     ,A(083)),(IY     ,A(084)),
     4            (IZ     ,A(085)),(IXY    ,A(086)),(IYZ    ,A(087)),
     5            (IXZ    ,A(088)),(XCG    ,A(611)),(ZCG    ,A(612))
     6           ,(DXCA,A(095)),(DYCA,A(096)),(DZCA,A(097)),
     7            (FXS    ,A(148)),(FYS    ,A(149)),(FZS    ,A(150)),
     8            (LS     ,A(151)),(MS     ,A(152)),(NS     ,A(153))
C
C                         **** FUNCTIONS ****
C
      EQUIVALENCE (CL     ,F(01)),(CZQ    ,F(02)),(CZALDT,F(03)),
     1            (CLDS   ,F(04)),(CLDSP  ,F(05)),(CLDA   ,F(06)),
     2            (CLDSB  ,F(07)),(CLDLG  ,F(08)),(CD     ,F(09)),
     3            (CDDSB  ,F(10)),(CDDLG  ,F(11)),(CMCA   ,F(12)),
     4            (CMQ    ,F(13)),(CMALDT,F(14)),(CMDS   ,F(15)),
     5            (CMDSP  ,F(16)),(CMDA   ,F(17)),(CMDSB  ,F(18)),
     6            (CMDLG  ,F(19)),(CYBET  ,F(20)),(CYR    ,F(21)),
     7            (CYP    ,F(22)),(CYDSP  ,F(23)),(CYDA   ,F(24)),
     8            (CYDR   ,F(25)),(CNBET  ,F(26)),(CNR    ,F(27)),
     9            (CNP    ,F(28)),(CNDSP  ,F(29)),(CNDA   ,F(30)),
     A            (CNDR   ,F(31)),(CLLBET,F(32)),(CLLR   ,F(33)),
```

Figure 20. Subroutine AERK Program Listing

```
     B              (CLLP   ,F(34)),(CLLDSP,F(35)),(CLLDA  ,F(36)),
     C              (CLLDR  ,F(37))
      REAL L,M,N,LS,MS,NS,IX,IY,IZ,IXY,IYZ,IXZ,MASS,MACH
      IF(MODE)100,200,500
  100 IF(MODE.LE.-2) RETURN
      PI=3.141592653589793
      PIC=PI/180.
      PICIN=1./PIC
      CD2=C*.5
      BD2=B*.5
      RNDM=0.
      RETURN
  200 CONTINUE
  500 CONTINUE
      SUMXT=XT(1)+XT(2)
      IF(SUMXT.GT.100.) GOTO 510
      POWER=0.
      GOTO 520
  510 POWER=1.
  520 CONTINUE
C
C     PERFORM LOOKUP
C
      V(2)=H
C     WEIGHTS AND MOMENT OF INERTIA
C
      CALL FLOOK(V,F,53,54)
      RHO=F(53)
      SOS=F(54)
      MASS=WT/G
      V(10)=YTRM
      V(15)=WT
      IF(RNDM)600,550,600
  550 CALL FLOOK(V,F,45,50)
      XCG=F(45)
      ZCG=F(46)
      ZCA=-ZCG
C
C  POSITION VECTOR OF A.C.
C
      DXCA =(XCG + XCA)/12.
      DYCA = YCA/12.
      DZCA =(ZCG + ZCA)/12.
      IX =F(47)
      IY =F(48)
      IZ =F(49)
      IXZ=F(50)
      IF (MACH .EQ. 0.0)  MACH = VEL / SOS
      IF (VEL .EQ. 0.0)  VEL = MACH * SOS
      VA=VSIDE-VG
      UAS=SQRT(VEL*VEL-VA*VA)
      UA=UAS*COS(AL)
      WA=UAS*SIN(AL)
      U=UA+UG
```

Figure 20. Subroutine AERK Program Listing (continued)

```
      VSIDE=VA+VG
      W=WA+WG
      RNDM=1.
  600 CONTINUE
      UA=U-UG
      VA=VSIDE-VG
      WA=W-WG
      PA=P-PG
      QA=Q-QG
      RA=R-RG
      UAS=SQRT(UA*UA+WA*WA)
      AL=ATAN2(WA,UA)
      ALD=AL*PICIN
      BET=ATAN2(VA,UAS)
      BETD=BET*PICIN
      VEL=SQRT(UA*UA+VA*VA+WA*WA)
      MACH=VEL/SOS
      ALDT=(UA*WDOT-WA*UDOT)/(UAS*UAS)
      V(1)=MACH
      V(3)=ALD+1.
      CALL FLOOK(V,F,1,1)
      V(4)=POWER
      V(5)=CL
      V(6)=YSB
      CALL FLOOK(V,F,2,37)
C
      DO 525 I=1,37
  525 A(800+I)=F(I)
      QBAR =.5*RHO*VEL*VEL
      QBARS=QBAR*S
C
C   COMPUTE TOTAL AERODYNAMIC COEFFICIENTS
C
      CAL=COS(AL)
      SAL=SIN(AL)
      UAS = CAL*UA+SAL*WA
      PAS = CAL*PA+SAL*RA
      QAS = QA
      RAS =-SAL*PA+CAL*RA
      CK1 = BD2/UAS
      CK2 = CD2/UAS
      CAD = CK2*ALDT
      CBD = CK1*BETD
      CP  = CK1*PAS
      CQ  = CK2*QAS
      CR  = CK1*RAS
      AYSP= ABS(YSP)
      AYDA= ABS(YDA)
C
      SUMCD = CD + CDDSB + CDDLG*YLG
C
      SUMCY = CYBET*BETD + CYP*CP + CYR*CR + CYDA*YDA + CYDR*YDR + CYDSP
     1*YSP
C
```

Figure 20. Subroutine AERK Program Listing (continued)

```
      SUMCL = CL - CZALDT*CAD - CZQ*CQ + CLDA*AYDA + CLDS*YDS + CLDSP*AY
     1SP + CLDSB*YSB + CLDLG*YLG
C
      SUMCLL= CLLBET*BETD + CLLP*CP + CLLR*CR + CLLDA*YDA + CLLDR*YDR +
     1CLLDSP*YSP
C
      SUMCMCA= CMCA + CMALDT*CAD + CMQ*CQ + CMDA*YDA + CMDS*YDS + CMDSP*
     1YSP + CMDSB*YSB + CMDLG*YLG
C
      SUMCN = CNBET*BETD + CNP*CP + CNR*CR + CNDA*YDA + CNDR*YDR + CNDSP
     1*YSP
C
C FORCES IN STABILITY AXES.
C
      FXS = -QBARS*SUMCD
      FYS =  QBARS*SUMCY
      FZS = -QBARS*SUMCL
C  TRANSFORM TO BODY AXES
C
C
      FX = FXS*CAL-FZS*SAL
      FY = FYS
      FZ = FXS*SAL+FZS*CAL
C
C  MOMENTS IN STABILITY AXES AT A.C.
C
      LS = QBARS*B*SUMCLL
      MS = QBARS*C*SUMCMCA
      NS = QBARS*B*SUMCN
C
C  MOMENTS IN BODY AXES AT C.G.
C
      L = LS*CAL-NS*SAL-DZCA*FY+DYCA*FZ
      M = MS+DZCA*FX-DXCA*FZ
      N = LS*SAL+NS*CAL-DYCA*FX+DXCA*FY
      RETURN
      END
```

Figure 20. Subroutine AERK Program Listing (concluded)

Table XII. List of Symbols for Subroutine AERK

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
α_a	AL	29		X		Angle of attack wrt air mass in radians
α_a°	AKD	706	deg	X		Angle of attack in degrees
$\dot{\alpha}_a$	ALDT	37	rad/sec	X		Derivative of angle of attack
b	B	115	ft			Wing span
β_a	BET	30	rad			Side slip angle
β_a°	BETD	707	deg		X	Side slip angle
$\dot{\beta}_a$	BETDT	38	rad/sec			Rate of change of side slip angle
c	C	116	ft			Wing mean aerodynamic chord
C_D	CD	809		X		Drag coefficient
$C_{D_{\delta_{lg}}}$	CDDLG	811		X		Drag coefficient for landing gear
$C_{D_{\delta_{sb}}}$	CDDSB	810		X		Drag coefficient for speed brakes
C_L	CL	801			X	Lift coefficient
$C_{L_{\delta_a}}$	CLDA	806	per deg		X	Lift coefficient of δ_a

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Manemonic	A-Array Index	Units	Input	Output	Description
$C_{L_{\delta_{lg}}}$	CLDLG	808			X	Lift coefficient of landing gear
$C_{L_{\delta_s}}$	CLDS	804	per deg		X	Lift coefficient of δ_s
$C_{L_{\delta_{sb}}}$	CLDSB	307	per deg		X	Lift coefficient of brakes
$C_{L_{\delta_{sp}}}$	CLDSP	805	per deg		X	Lift coefficient of δ_{sp}
C_{ℓ_β}	CLLBET	832	per deg		X	Rolling moment coefficient of β
$C_{\ell_{\delta_a}}$	CLLDA	836	per deg		X	Rolling moment coefficient of δ_a
$C_{\ell_{\delta_r}}$	CLLDR	837	per rad		X	Rolling moment coefficient of δ_r
$C_{\ell_{\delta_{sp}}}$	CLLDSP	835	per deg		X	Rolling moment coefficient of δ_{sp}
C_{ℓ_p}	CLLP	834	per rad/sec		X	Rolling moment coefficient of p
C_{ℓ_r}	CLLR	83[illegible]	per rad/sec		X	Rolling moment coefficient or r
C_{mCa}	CMCA	8 12			X	Pitching moment coefficient
$C_{m\dot{\alpha}}$	CMALDT	839	per rad/sec		X	Pitching moment coefficient of α

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
$C_{m_{\delta_{lg}}}$	CMDLG	819			X	Pitching moment coefficient of landing gear
$C_{m_{\delta_s}}$	CMDS	815	per deg		X	Pitching moment coefficient of δ_s
$C_{m_{\delta_{sb}}}$	CMDSB	818	per deg		X	Pitching moment coefficient of speed brakes
$C_{m_{\delta_{sp}}}$	CMDSP	816	per deg		X	Pitching moment coefficient of δ_{sp}
C_{m_q}	CMQ	813	per rad/sec		X	Pitching moment coefficient of q
C_{n_β}	CNBET	826	per rad		X	Yawing moment coefficient of β
$C_{n_{\delta_a}}$	CNDA	830	per deg		X	Yawing moment coefficient of δ_a
$C_{n_{\delta_r}}$	CNDR	831	per rad		X	Yawing moment coefficient of δ_r
$C_{n_{\delta_{sp}}}$	CNDSP	829	per deg		X	Yawing moment coefficient of δ_{sp}
C_{n_p}	CNP	828	per rad/sec		X	Yawing moment coefficient of p
C_{n_r}	CNR	827	per rad/sec		X	Yawing moment coefficient cf r
C_{Y_β}	CYBET	820	per deg		X	Side force coefficient of β

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
$C_{Y_{\delta_a}}$	GYDA	824	per deg		X	Side force coefficient of δ_a
$C_{Y_{\delta_r}}$	CYDR	825	per rad		X	Side force coefficient of δ_r
$C_{Y_{\delta_{sp}}}$	CYDSP	823	per deg		X	Side force coefficient of δ_{sp}
C_{Y_p}	CYP	822	per rad/sec		X	Side force Coefficient of p
C_{Y_r}	CYR	821	per rad/sec		X	Side force coefficient of r
$C_{z_{\dot{\alpha}}}$	CSALDT	803	per rad/sec		X	Side force coefficient of $\dot{\alpha}$
C_{z_q}	CZQ	802	per rad/sec		X	Side force coefficient of q
$\Delta_{x_{ca}}$	DXCA	95	inches		X	Components of moment reference center distance vector from c.g
$\Delta_{y_{ca}}$	DYCA	96	inches		X	
$\Delta_{z_{ca}}$	DZCA	97	inches		X	
X	FX	71	lbs			Aerodynamic force component along body x-axes
X_S	FXS	148	lbs			Aerodynamic force component along stability x-axis
Y	FY	72	lbs		X	Aerodynamic force component along body y-axis

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
Y_s	FYS	149	lbs		X	Aerodynamic force component along stability y-axis
Z	FZ	73	lbs		X	Aerodynamic force component along body z-axis
Z_s	FZS	150	lbs		X	Aerodynamic force component along stability z-axis
g	G	90	ft/sec^2	X		Acceleration of gravity
γ	GAM	28	rad			Flight path angle
h	H	4	ft			Altitude of A/C c.g
I_x	IX	83	slug-ft^2		X	
I_{xy}	IXY	86	slug-ft^2		X	
I_{xz}	IXZ	88	slug-ft^2		X	Moments and cross-products of inertia of the body about body axes located at cg
I_y	IY	84	slug-ft^2		X	
I_{yz}	IYZ	87	slug-ft^2		X	
I_z	IZ	85	slug-ft^2		X	
L	L	77	ft-lbs		X	Aerodynamic moment about body x-axis
L_s	LS	151	ft-lbs		X	Aerodynamic moment about stability x-axis
M	M	78	ft-lbs		X	Aerodynamic moment about body y-axis
Mach	MACH	5		X		Mach number

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
m	MASS	91	slug		X	Aircraft mass
M_s	MS	152	ft-lbs		X	Aerodynamic moment about stability y-axis
N	N	79	ft-lbs		X	Aerodynamic moment about body z-axis
N_s	NS	153	ft-lbs		X	Aerodynamic moment about stability z-axis
p_a	PA		rad/sec		X	Roll rate wrt air mass
p_{as}	PAS		rad/sec		X	Roll rate in stability axes
π	PI		rad		X	Constants
$\pi/180$	PIC		rad/deg		X	Constants
$(\pi/180)^{-1}$	PICIN		deg/rad		X	Constants
Pow	POWER		percent		X	Flag for thrust level (0 or 1)
q_a	QA		rad/sec		X	Pitch rate wrt air mass
q_{as}	QAS		rad/sec		X	Pitch rate in stability axes
$\bar{q}$	QBAR	70	lb/ft^2		X	Dynamic pressure
$\bar{q}S$	QBARS		lb		X	Force coefficient
r_a	RA		rad/sec		X	Yaw rate wrt air mass
r_{as}	RAS		rad/sec		X	Yaw rate in stability axes

Table XII. List of Symbols for Subroutine AERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
ρ	RHO	114, 853	lbs/ft^3		X	Air density
	RNDM				X	First-time flag (initially zero)
S	S	117	ft^2	X		Wing surface area
	SAC			X		Aircraft simulation flat (see p. 34)
a	SOS	6, 854	ft/sec		X	Speed of sound
ΣC_D	SUMCD				X	Total drag coefficient
ΣC_L	SUMCL				X	Total lift coefficient
ΣC_ℓ	SUMCLL				X	Total rolling moment coefficient
ΣC_m	SUMCMCA				X	Total pitching moment coefficient at moment reference center
ΣC_n	SUMCN				X	Total yawing moment coefficient
ΣC_y	SUMCY				X	Total side force coefficient
ΣX_T	SUMXT		percent		X	Total percent thrust output of engines 1 and 2
u_a	UA		ft/sec		X	Component of velocity wrt air mass along body x-axis
u_{as}	UAS		ft/sec		X	Component of velocity wrt air mass along stability x-axis
v	V		ft/sec		X	Component of velocity wrt air mass along body y-axis

Table XII. List of Symbols for Subroutine AERK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
V	VEL		ft/sec		X	Magnitude of velocity vector wrt air mass
w_a	WA		ft/sec		X	Component of velocity wrt air mass along body z-axis
wt	WT		lbs		X	Weight of aircraft
x_{ca}	XCA	579	inches		X	x coordinate of moment center from body fixed reference point
x_{cg}	XCG	611	inches		X	x coordinate of cg from body fixed reference point
$X_T(1)$	XT(1)	559	percent		X	Thrust output of engine 1
y_{ca}	YCA	580	inches		X	y coordinate of moment center from body fixed reference point
$y_{\delta a}$	YDA	123	deg	X		Effective aileron deflection
$y_{\delta r}$	YDR	125	rad	X		Effective rudder deflection
$y_{\delta s}$	YDS	121	deg	X		Effective stabilator deflection
$y_{\ell g}$	YLG	648		X		Effective landing gear deflection
y_{sb}	YSB	646	deg	X		Effective speed brake deflection
y_{sp}	YSP	645	deg	X		Effective spoiler deflection
y_{trim}	YTRM	651	deg	X		Aileron trim deflection
z_{ca}	ZCA	581	inches		X	z component of moment center from cg
z_{cg}	ZCG	612	inches		X	z coordinate of moment center from body fixed reference point

Table XIII. Representation of Bomb Aerodynamic Coefficients in Cross-Velocity Axes

Coefficient	Look-Up Representation	Mnemonic	Units
$C_N(\hat{\alpha}^o, M_a)$	F75(3, 1)	CN	
$C_{N_\delta}(\hat{\alpha}^o, M_a)$	F76(3, 1)	CNDEL	per degree
$C_A(M_a)$	F77(1)	CA	
$C_m(\hat{\alpha}^o, M_a)$	F78(3, 1)	CM	
$C_{m_q}(\hat{\alpha}^o, M_a)$	F79(3, 1)	CMQ	per degree
$C_{m_\delta}(\hat{\alpha}^o, M_a)$	F80(3, 1)	CMDEL	per degree

Argument	Look-Up Representation	Mnemonic	Units
M	V(1)	MACH	
h	V(2)	H	feet
$\hat{\alpha}^o$	V(3)	ALFH	degrees

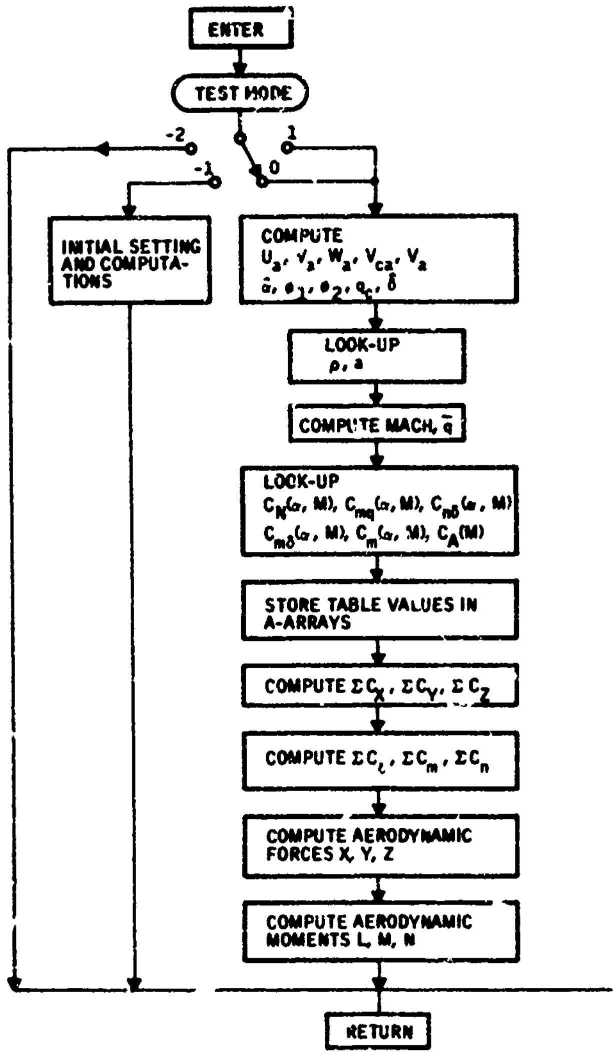


Figure 21. Subroutine WAERK Flow Diagram

```
      SUBROUTINE WAERK
      COMMON/ADAP/MODE,A(1000)
      DIMENSION VST(20),F(80)
      EQUIVALENCE (U      ,A(007)),(V      ,A(008)),(W      ,A(009)),
     A            (UG     ,A(101)),(VG     ,A(102)),(WG     ,A(103)),
     1                             (Q      ,A(043)),(R      ,A(044)),
     2            (BMBD   ,A(294)),(DELTAZ,A(263)),(DELTAY,A(264)),
     3            (H      ,A(004)),(RHO    ,A(114)),(MACH   ,A(005)),
     4            (SOS    ,A(006)),(QBAR   ,A(070)),(L      ,A(077)),
     5            (M      ,A(078)),(N      ,A(079)),(X      ,A(071)),
     6            (Y      ,A(072)),(Z      ,A(073))
      REAL L,M,N,MACH
      IF(MODE)100,200,200
  100 IF(MODE.LE.-2)RETURN
      PI=3.141592653589793
      PIC=PI/180.
      PICIN=1./PIC
      BS=PI*BMBD*BMBD*.25
      Q=.0000001
      RETURN
  200 CONTINUE
      UA=U-UG
      VA=V-VG
      WA=W-WG
      VCA=SQRT(VA*VA+WA*WA)
      DELTAH=SQRT(DELTAZ*DELTAZ+DELTAY*DELTAY)
      VTA=SQRT(VCA*VCA+UA*UA)
      ALFH  =ATAN2(VCA,UA)
      PHI=ATAN2(VA,WA)
      CPHI=COS(PHI)
      SPHI=SIN(PHI)
      PHI2=ATAN2(-R,Q)
      CPHI2=COS(PHI2)
      SPHI2=SIN(PHI2)
      QC=SQRT(Q*Q +R*R)
      ALFHD=ALFH*PICIN
      XYZ=BMBD*QC*.5/VTA
      VST(2)=H
      CALL FLOOK(VST,F,53,54)
      RHO=F(53)
      SOS=F(54)
      MACH=VTA/SOS
      QBAR=.5*RHO*VTA*VTA
      VST(1)=MACH
C TAKE OUT
      SIGN=1.
      IF(ALFHD.LT.0.) SIGN=-1.
      IF(ALFHD.LT.-4.) ALFHD=-4.
      VST(3)=ABS(ALFHD)
      CALL FLOOK(VST,F,75,80)
      CN=SIGN*F(75)
      CMQ=F(79)
      CNDEL=SIGN*F(76)
      CMDEL=SIGN*F(80)
```

Figure 22. Subroutine WAERK Program Listing

```
      CM=SIGN*F(78)
C TAKE OUT
      CA    =F(77)
      A(875)=CN
      A(876)=CNDEL
      A(877)=CA
      A(878)=CM
      A(879)=CMQ
      A(880)=CMDEL
      A(881)=PHI2
      A(882)=PHI
      SUMCX=-CA
       SUMCY=SPHI*(-CN-CNDEL*DELTAH)
      SUMCZ=CPHI*(-CN-CNDEL*DELTAH)
      SUMCL=0.
      SUMCM=CPHI*(CM+CMDEL*DELTAH)+CPHI2*CMQ*XYZ
      SUMCN=-SPHI*(CM+CMDEL*DELTAH)-SPHI2*CMQ*XYZ
      QBARS=QBAR*BS
      QBARSD=QBARS*BMBD
      X=QBARS*SUMCX
      Y=QBARS*SUMCY
      Z=QBARS*SUMCZ
      L=QBARSD*SUMCL
      M=QBARSD*SUMCM
      N=QBARSD*SUMCN
      RETURN
      END
```

Figure 22. Subroutine WAERK Program Listing (concluded)

Table XIV. List of Symbols for Subroutine WAERK

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
$\hat{\alpha}$	ALFH		rad		X	Magnitude of yaw
$\hat{\alpha}°$	ALFHD		deg		X	Magnitude of yaw
d_b	BMBD	294	ft	X		Bomb diameter
S_b	BS		ft^2		X	Bomb corss-section area
C_A	CA	877			X	Axial force coefficient along cross velocity x_1 axis
C_m	CM	878			X	Restoring moment coefficient about cross velocity y_1 axis
C_{m_δ}	CMDEL	880	per deg		X	Restoring moment coefficient of canted fin
C_{m_q}	CMQ	879	per deg/sec		X	Damping moment coefficient about y_1 axis
C_N	CN	875			X	Normal force coefficient
C_{N_δ}	CNDEL	876	per deg		X	Coefficient of normal force due to canted fin
$\hat{\delta}$	DELTAH		deg		X	Magnitude of first cant angle
δ_y	DELTAY	264	deg	X		Fin cant angle about y axis
δ_z	DELTAZ	263	deg	X		Finc ant angle about z axis
	H	4	ft	X		Altitude of cg
L	L	77	ft-lb		X	Rolling moment about x-axis

Table XIV. List of Symbols for Subroutine WAERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
M	M	78	ft-lb		X	Pitching moment about y-axis
	MACH	5			X	Mach number
N	N	79	ft-lb		X	Yawing moment about z-axis
ϕ	PHI	882	rad		X	Rotation about x axis from z to z_1
ϕ_2		881	rad		X	Rotation about x axis from y to y_1
π	PI				X	Constants
$\pi/180$	PIC				X	Constants
$180/\pi$	PICIN				X	Constants
q	Q	43	deg/sec	X		Pitch rate about y
$\bar{q}$	QBAR	70	lbs/ft^2		X	Dynamic pressure
$\bar{q}Sd$	QBARSD		ft-lbs		X	Moment coefficient
q_c	QC		deg/sec		X	Pitch rate about cross-spin axis y_2
r	R	44	deg/sec	X		Yaw rate about z
ρ	RHO	114	lbs/ft^3		X	Air density
	SIGN				X	Factor for extending the range of table
a	SOS	6	ft/sec		X	Speed of sound

Table XIV. List of Symbols for Subroutine WAERK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
ΣC_L	SUMCL				X	Total rolling moment coefficient about x axis
ΣC_m	SUMCM				X	Total pitching moment coefficient about y axis
ΣC_N	SUMCN				X	Total yawing moment coefficient about z axis
ΣC_x	SUMCX				X	Total force coefficient along x axis
ΣC_y	SUMCY				X	Total force coefficient along y axis
ΣC_z	SUMCZ				X	Total force coefficient along z axis
u	U	7	ft/sec	X		Inertial velocity component along x axis
u_a	UA		ft/sec		X	Component of velocity wrt air mass along x axis
u_g	UG	101	ft/sec	X		Component of gust velocity along x axis
v	V	8	ft/sec	X		Inertial velocity along x axis
v_a	VA		ft/sec		X	Component of velocity wrt air mass along y axis
v_{ca}	VCA		ft/sec		X	Cross velocity
v_g	VG	102	ft/sec	X		Component of gust velocity along y axis

Table XIV. List of Symbols for Subroutine WAERK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
	VST(I)				X	Table lookup variable
	VTA		ft/sec		X	Magnitude of velocity of cg wrt air mass
w	W	9	ft/sec	X		Inertial velocity along z axis
w_a	WA		ft/sec		X	Component of velocity wrt air mass along z axis
w_g	WG	103	ft/sec	X		Component of gust velocity along z axis
X	X	71	lbs		X	Force component along x axis
Y	Y	72	lbs		X	Force component along y axis
Z	Z	73	lbs		X	Force component along z axis

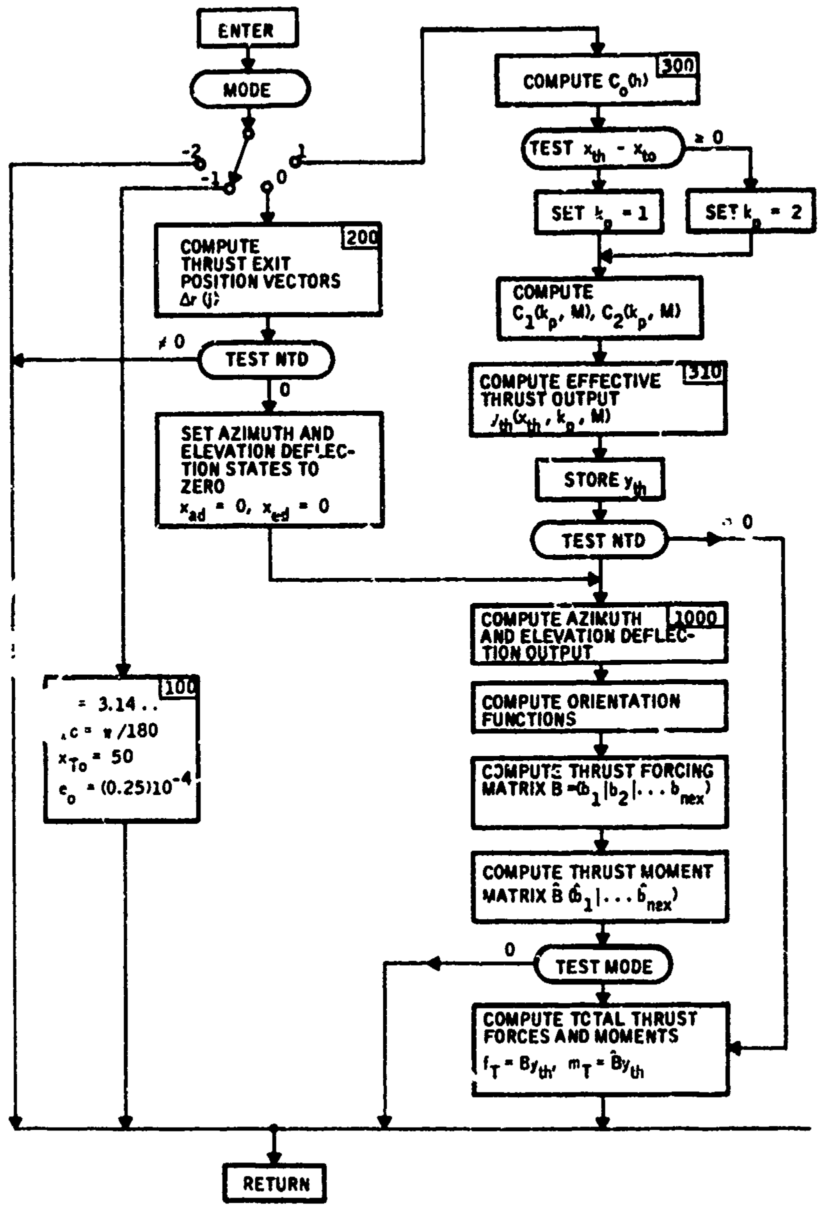


Figure 23. Subroutine THRUSK Flow Diagram

```
      SUBROUTINE THRUSK
      COMMON/ADAP/MODE,A(1000)
C
      DIMENSION XEX(5),YEX(5),ZEX(5),DXEX(5),DYEX(5),DZEX(5),XTH(5),
     1XAD(5),XED(5),YTH(5),YAD(5),YED(5),YAB(5),YEB(5),YTB(2,5),CTH(2,5)
     2,E1(2,5),E2(2,5),B(3,5),BH(3,5)
C
C                 *** PARAMETER INPUTS ***
C
      EQUIVALENCE  (XEX(1)  ,A(501)),(YEX(1)  ,A(506)),(ZEX(1)  ,A(511))
     1             ,(XCG     ,A(611)),(ZCG     ,A(612)),(ANTD    ,A(516))
     2             ,(ANEX    ,A(517)),(YTB(1,1),A(518)),(CTH(1,1),A(528))
     3             ,(E1(1,1),A(538)),(E2(1,1),A(548)),(EO       ,A(558))
     4             ,(YAB(1),A(582)),(YEB(1),A(587))
C
C                 *** VARIABLE  INPUTS ***
C
      EQUIVALENCE  (H       ,A(004)),(MACH    ,A(005)),(XTH(1)  ,A(559))
     4             ,(XAD(1)  ,A(564)),(XED(1)  ,A(569))
C
C                 *** VARIABLE OUTPUTS ***
C
      EQUIVALENCE  (XT      ,A(111)),(ZT      ,A(112)),(YT      ,A(113))
     1             ,(LT      ,A(600)),(MT      ,A(601)),(NT      ,A(602))
     2             ,(YTH(1),A(574))
      REAL LT,MT,NT,MACH
      IF(MODE)100,200,300
  100 IF(MODE.LE.-2) RETURN
      PI=3.141592653589793
      PIC=PI/180.
      XTO=50.
      EO =.000025
      NTD=ANTD
      NEX=ANEX
      RETURN
  200 DO 210 J=1,NEX
      DXEX(J)=(XCG+XEX(J))/12.
      DYEX(J)=(YEX(J))/12.
  210 DZEX(J)=(ZCG+ZEX(J))/12.
      A(092)=DXEX(1)
      A(093)=DZEX(1)
      IF(NTD.NE.0) RETURN
      DO 220 J=1,NEX
      XAD(J)=0.
  220 XED(J)=0.
      GOTO 1000
  300 CO=1.+EO*H
      DO 310 J=1,NEX
      IF(XTH(J)-XTO)302,303,303
  302 KP=1
      GOTO 304
  303 KP=2
  304 C1=1.+E1(KP,J)*MACH
      C2=1.+E2(KP,J)*MACH
```

Figure 24. Subroutine THRUSK Program Listing

```
 310 YTH(J)=C0*(C1*YTU(KP,J)+C2*[illegible]H(KP,J)*(XTH(J)-XTO))
     YTH(1) = A(141)
     YTH(2) = A(142)
     IF(NTD.EQ.0) GOTO 200[illegible]
1000 DO 1010 J=1,N[illegible]X
     YAD(J)=YAR(J)+XAD(J)
     YED(J)=YEB(J)+XED(J)
     RAD=YAD(J)*PIC
     RFD=YFD(J)*PIC
     CYAD=COS(RAD)
     SYAD=SIN(RAD)
     CYE[illegible]COS(RFD)
     SYE[illegible]IN(RE[illegible])
     B([illegible])=CYAD*C[illegible]E[illegible]
     B([illegible])SYA[illegible]
1010 B(3,[illegible]CYA[illegible]SYE[illegible]
     DO [illegible] J=1 [illegible]X
     BH(1[illegible])=DYEX(J[illegible])-DZEX(J)*B(2,J)
     BH(2,J[illegible]DZEX(J[illegible]J)-DXEX(J)*B(3,J)
1020 BH(3,J)=DXEX(J[illegible],J)-DYEX(J)*B(1,J)
     A(593)=B(1,1)
     A(594)=B(3,1)
     A(595)=BH(2,1)
     A(596) = B(1,2)
     A(597) = B(3,2)
     A(598) = BH(2,2)
     YTH(1)=A(141)
     YTH(2)=A(142)
     IF(MODE.EQ.0) RETURN
2000 XT=0.
     YT=0.
     ZT=0.
     LT=0.
     MT=0.
     NT=0.
     DO 2010 J=1,NEX
     XT=XT+B(1,J)*YTH(J)
     YT=YT+B(2,J)*YTH(J)
     ZT=ZT+B(3,J)*YTH(J)
     LT=LT+BH(1,J)*YTH(J)
     MT=MT+BH(2,J)*YTH(J)
2010 NT=NT+BH(3,J)*YTH(J)
     RETURN
     END
```

Figure 24. Subroutine THRUSK Program Listing (concluded)

Table XV. List of Symbols for Subroutine THRUSK

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
n_{ex}	ANEX, NEX	517		X		Number of thrust exits
n_{td}	ANTD, NTD	516		X		Number of thrust deflectors
B	B(1, 1)	593				Thrust force coefficient matrix
$\hat{B}$	BH(I, J)					Thrust moment coefficient matrix
c_1	C1				X	
c_2	C2				X	Thrust effectiveness factors see p. 56 of Vol. I
c_0	C0				X	
c	CTH(1, 1)	528		X		Thrust output coefficient
Δx_{ex}	DXEX(1)	92	ft			
Δy_{ex}	DYEX(1)		ft			Components of the vector from cg to thrust exit 1
Δz_{ex}	DZEX(1)	93	ft			
e_1	E1(1, 1)	538		X		
e_2	E2(1, 1)	548		X		Thrust effectiveness coefficients
e_0	E0	558		X		
h	H	4	ft	X		Altitude
L_T	LT	600	lbs		X	Thrust moment about x axis
M	MACH	5		X		Mach number

Table XV. List of Symbols for Subroutine THRUSK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
M_T	MT	601	ft-lbs		X	Thrust moment about y axis
N_T	NT	602	ft-lbs		X	Thrust moment about z axis
π	PI					Constants
$\pi/180$	PIC					Constants
ψ	RAD		rad			Effective azimuth angle of nozzle axis
θ	RED		rad			Effective elevation angle of nozzle axis
ψ_{T1}	XAD(1)	564	deg	X		Azimuth angle state of nozzle axis no. 1
x_{cg}	XCG	611		X		x component of the vector from cg to RP
θ_{T1}	XED(1)	569	deg	X		Elevation angle state of nozzle axis no. 1
x_{ex}	XEX(1)	501		X		x component of the vector from RP to thrust exit 1
X_T	XT	111			X	Thrust force along x axis
x_{Th1}	XTH(1)	559	percent	X		Thrust magnitude state of engine no. 1
ξ_T	XT0					Magnitude dependent coefficient
ψ_{b1}	YAB(1)	582	deg	X		Effective azimuth bias

Table XV. List of Symbols for Subroutine THRUSK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
θ_{b1}	YEB(1)	587	deg	X		Effective elevation bias
$\psi^{\circ}(1)$	YAD(1)		deg		X	Effective azimuth angle of nozzle axis no. 1
$\theta^{\circ}(1)$	YED(1)		deg		X	Effective elevation angle of nozzle axis no. 1
y_{ex}	YEX(1)	506		X		y component of the vector from RP to thrust exit 1
Y_T	Y_T	113	lbs		X	Thrust force along y axis
y_o	YTB(1, 1)	518		X		Thrust bias
T_1	YTH(1)*	574	lbs		X	Effective thrust magnitude of engine 1
z_{cg}	ZCG	612	in.	X		z component of the vector from cg to RP
z_{ex}	ZEX(1)	511	in.	X		z component of the vector from RP to exit 1
Z_T	ZT	112	lbs		X	Thrust force along z axis

*In the present program, T_1 and T_2 are set to the values of A(141) and A(142), respectively.

Subroutine WINDK

Subroutine WINDK implements the model developed in Section IV of Volume I. It generates the mean-wind and the wind-gust velocities along the aircraft and weapon body axes.

The subroutine WINDK flow diagram is shown in Figure 25 a d the program listing in Figure 26. Symbols are listed in Table XVI.

Subroutine SENK

Subroutine SENK implements the model developed in Section V of Volume I. It generates the sensed-output signals at various points on the aircraft in terms of aircraft states and their derivatives.

The subroutine SENK flow diagram is shown in Figure 27 and the program listing in Figure 28.

Subroutine PILOT

Subroutine PILOT implements the model developed in Section VII of Volume I. It generates the stabilator (i.e., elevator) signal to keep the aircraft along a dive and/or pull-up path.

The subroutine PILOT flow diagram is shown in Figure 29 and the program listing in Figure 30.

Subroutine NOMK

Subroutine NOMK is assigned to generate the nominal (trim) parameters by algebraic approach developed in Section VII of Volume I. It has not been programmed in this work (trimming is done by PILOT). This subroutine when programmed should provide more accurate values for the nominal trajectory and the trim profile. The subroutine NOMK flow diagram is shown in Figure 31.

Subroutine RELK

Subroutine RELK generates the nominal release time of the weapon for the dive and pull-up maneuver as described in Appendix II of Volume I. Since the weapon release model depends heavily on the fire control system used in the aircraft, it must be written separately by the user. The subroutine RELK flow diagram is shown in Figure 32.

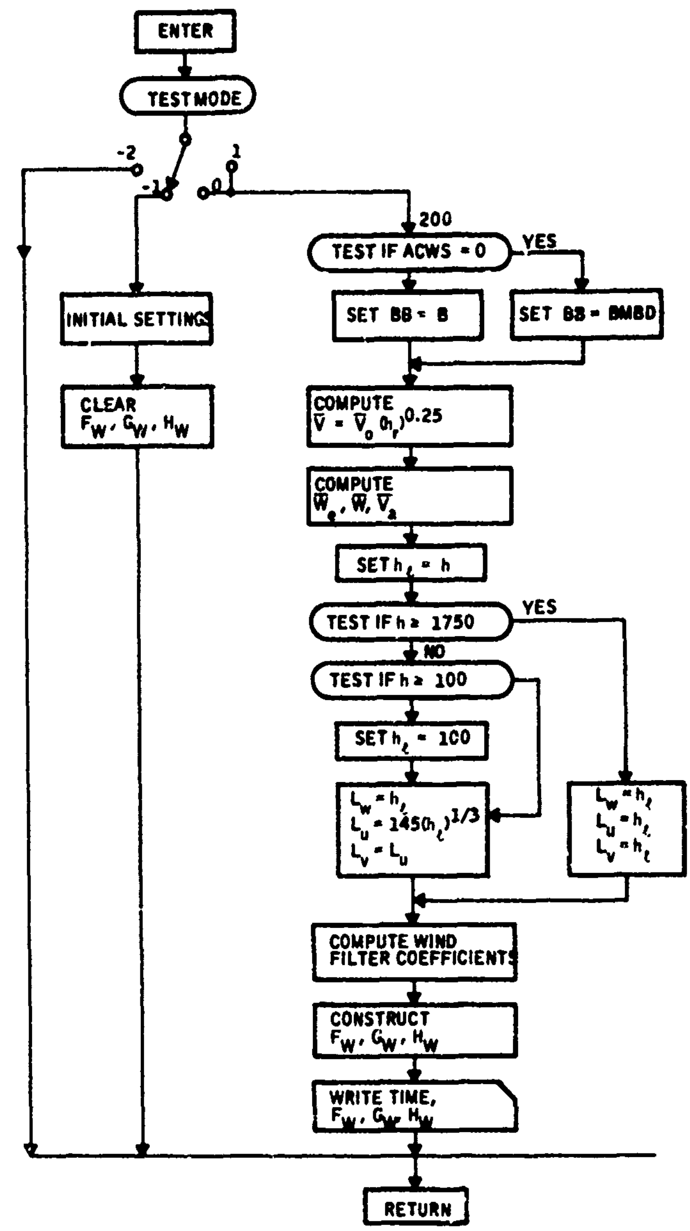


Figure 25. Subroutine WINDK Flow Diagram

```
      SUBROUTINE WINDK(FW,GW,HW)
      COMMON/ADAP/MODE,A(1000)
      DIMENSION FW(8,8),GW(8,4),HW(6,8)
      EQUIVALENCE (U      ,A(007)),(V      ,A(008)),(W      ,A(009)),
     1            (PSIB   ,A(290)),(THETAB,A(291)),(H      ,A(004)),
     2            (HNAUT  ,A(292)),(VBNAUT,A(293)),(E11    ,A(054)),
     3            (E21    ,A(055)),(E31    ,A(056)),(E12    ,A(057)),
     4            (E22    ,A(058)),(E32    ,A(059)),(E13    ,A(060)),
     5            (E23    ,A(061)),(E33    ,A(062)),(B      ,A(115)),
     6            (BMBD   ,A(294))
      EQUIVALENCE (UB     ,A(295)),(VB     ,A(296)),(WB     ,A(297)),
     1            (UBA    ,A(298)),(VBA    ,A(299)),(WBA    ,A(300)),
     2            (VBARA  ,A(301)),(SIGU   ,A(302)),(SIGV   ,A(303)),
     3            (SIGW   ,A(304)),(AU     ,A(305)),(AV     ,A(306)),
     4            (AW     ,A(307)),(AP     ,A(308)),(AQ     ,A(309)),
     5            (AR     ,A(310)),(VELB,A(288))
      REAL LW,LU,LV
      IF(MODE)100,200,200
  100 IF(MODE.LE.-2) RETURN
      PI=3.14159265
      LWR=9
      DO 101 I=1,8
      DO 102 J=1,8
  102 FW(I,J)=0.
      DO 103 J=1,4
  103 GW(I,J)=0.
      DO 101 J=1,6
  101 HW(J,I)=0.
      EX3=1./3.
      RETURN
  200 IF(A(998).EQ.0.) GOTO 204
      BB=B
      GOTO 205
  204 BB=BMBD
  205 HR=H/HNAUT
      VELB=VBNAUT*(HR)**.25
      CPB=COS(PSIB)
      SPB=SIN(PSIB)
      CTB=COS(THETAB)
      STB=SIN(THETAB)
      C1=CTB*CPB*VELB
      C2=CTB*SPB*VELB
      C3=-STB*VELB
      UB=E11*C1+E12*C2+E13*C3
      VB=E21*C1+E22*C2+E23*C3
      WB=E31*C1+E32*C2+E33*C3
      UBA=U-UB
      VBA=V-VB
      WBA=W-WB
      VBARA=SQRT(UBA*UBA+VBA*VBA+WBA*WBA)
      HL=H
      IF(H.GE.1750.) GOTO 201
      IF(H.GE.100.) GOTO 210
      HL=100.
```

Figure 26. Subroutine WINDK Program Listing

```
210 LW=HL
    LU=145.*(HL)**EX3
    LV=LU
    GOTO 202
201 HL=1750
    LW=HL
    LU=HL
    LV=HL
202 CONTINUE
    SIGW=10.25-1.25*ALOG10(HL)
    SIGU=SQRT(LU/LW)*SIGW
    SIGV=SQRT(LV/LW)*SIGW
    AU=VBARA/LU
    AV=VBARA/LV
    AW=VBARA/LW
    AP=VBARA*PI/(BB*4.)
    AQ=AP
    AR=VBARA*PI/(BB*3.)
    BU=SIGU*SQRT(2.*AU/PI)
    B2V=SIGV*SQRT(AV*3./PI)
    B1V=AV*B2V/SQRT(3.)
    A1V=AV*AV
    A2V=2.*AV
    B2W=SIGW*SQRT(AW*3./PI)
    B1W=AW*B2W/SQRT(3.)
    A1W=AW*AW
    A2W=2.*AW
    XYZ=(PI*LW/(BB*4.))**EX3
    XYZ=SQRT(.8*XYZ)*AP
    BP=SIGW*SQRT(1./(LW*VBARA))*XYZ
    DQ=AQ/VBARA
    BQ=AQ*DQ
    DR=-AR/VBARA
    BR=AR*DR
    FW(1,1)=-AU
    FW(2,2)=-A2V
    FW(2,4)=1.
    FW(3,3)=-A2W
    FW(3,5)=1.
    FW(4,2)=-A1V
    FW(5,3)=-A1W
    FW(6,6)=-AP
    FW(7,7)=-AQ
    FW(8,8)=-AR
    FW(7,3)=BQ
    FW(8,2)=BR
    GW(1,1)=BU
    GW(2,2)=B2V
    GW(3,3)=B2W
    GW(4,2)=B1V
    GW(5,3)=B1W
    GW(6,4)=BP
    HW(5,3)=DQ
    HW(6,2)=DR
    HW(1,1)=1.
    HW(2,2)=1.
    HW(3,3)=1.
    HW(4,6)=1.
    HW(5,7)=1.
    HW(6,8)=1.
302 FORMAT(//)
    WRITE(LWR,300)A(1)
300 FORMAT(1H1/7X,6H TIME=F12.5/)
    WRITE(LWR,301)
301 FORMAT(/18H MATRICES FW,GW,HW//)
    CALL MP(8,8,8,8,FW,LWR)
    WRITE(LWR,302)
    CALL MP(8,4,8,4,GW,LWR)
    WRITE(LWR,302)
    CALL MP(6,8,6,8,HW,LWR)
    RETURN
    END
```

Figure 26. Subroutine WINDK Program Listing (concluded)

Table XVI. List of Symbols for Subroutine WINDK

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
a_{1v}	A1V				X	Feedback element in v_g filter
a_{1w}	A1W				X	Feedback element in w_g filter
a_{2v}	A2V				X	Feedback element in v_g filter
a_{2w}	A2W				X	Feedback element in w_g filter
a_p	AP	308			X	Feedback element in p_g filter
a_q	AQ	309			X	Feedback element in q_g filter
a_r	AR	310			X	Feedback element in r_g filter
a_u	AU	305			X	Feedback element in u_g filter
a_v	AV	306			X	Auxiliary variables
a_w	AW	307			X	
b	B	115	ft	X		Wing span
b_{1u}	B1U				X	Input element in u_g filter
b_{1v}	B1V				X	Input element in v_g filter
b_{1w}	B1W				X	Input element in w_g filter
b_{2v}	B2V				X	Input element in v_g filter
b_{2w}	B2W				X	Input element in w_g filter
b_b	BB				X	Equivalent span

Table XVI. List of Symbols for Subroutine WINDK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
d_b	BMBD	294	ft	X		Bomb diameter
b_p	BP				X	Input element in p_g filter
b_q	BQ				X	Input element in q_g filter
b_r	BR				X	Input element in r_g filter
b_u	BU				X	Input element in u_g filter
$\bar{w}_1$	C1		ft/sec		X	Mean wind components along earth axes x_e, y_e, and z_e, respectively
$\bar{w}_2$	C2		ft/sec		X	
$\bar{w}_3$	C3		ft/sec		X	
d_q	DQ				X	Output element in q_g filter
d_r	DR				X	Output element in r_g filter
e_{11}	E11	54		X		Elements of the transformation matrix from earth to body axes
e_{21}	E21	55		X		
e_{31}	E31	56		X		
e_{12}	E12	57		X		
e_{22}	E22	58		X		
e_{32}	E32	59		X		
e_{13}	E13	60		X		
e_{23}	E23	61		X		
e_{33}	E33	62		X		

Table XVI. List of Symbols for Subroutine WINDK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
F_w	FW				X	Transition matrix of wind state
G_w	GW				X	Input matrix
h	H	4	ft	X		Altitude
h_o	HNAUT	292	ft	X		Reference altitude for mean wind
H_w	HW				X	Output matrix
	ITAPE			X		Logical tape number for time-varying coefficient data
L_u	LU		ft		X	Scale along x-axis
L_v	LV		ft		X	Scale along y-axis
L_w	LW		ft		X	Scale along z-axis
π	PI		rad	X		
$\bar{\psi}$	PSIB	290	rad	X		Azimuth angle of mean wind wrt earth axis
σ_u	SIGU	302	ft/sec		X	Root mean square gust velocities along x, y, z body axes
σ_v	SIGV	303	ft/sec		X	
σ_w	SIGW	304	ft/sec		X	
$\bar{\theta}$	THETAB	291	rad	X		Elevation angle of mean wind wrt earth axis

Table XVI. List of Symbols for Subroutine WINDK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
u	U	7	ft/sec	X		Velocity component of cg along x axis wrt earth
u_b	UB	295	ft/sec		X	Mean wind component along body x axis wrt earth
$\bar{u}_a$	UBA	298	ft/sec		X	Same wrt air mass
v	V	8	ft/sec		X	Velocity component of cg along y axis wrt earth
v_b	VB	296	ft/sec		X	Mean wind component along body y axis wrt earth
$\bar{v}_a$	VBA	299	ft/sec		X	Same wrt air mass
$\bar{V}_a$	VBARA	301	ft/sec		X	Mean wind speed wrt air mass
$\bar{V}_{bo}$	VBNAUT	293	ft/sec		X	Mean wind speed at reference altitude
$\bar{V}$	VELB	288	ft/sec	X	X	Mean wind magnitude
w	W	9	ft/sec		X	Velocity component of cg along z axis wrt earth
w_b	WB	297	ft/sec		X	Mean wind component along body z axis wrt earth
$\bar{w}_a$	WBA	300	ft/sec		X	Same wrt air mass

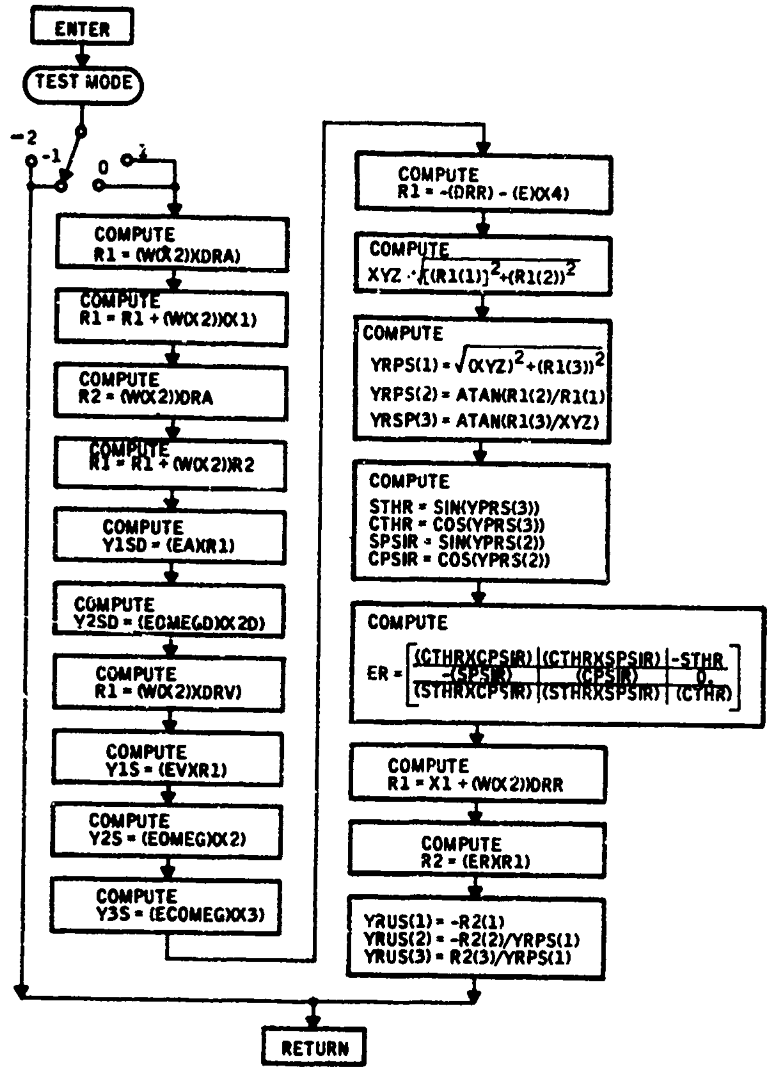


Figure 27. Subroutine SENK Flow Diagram

```
      SUBROUTINE SENK
      COMMON/ADAP/MODE,A(1000)
      DIMENSION EA(3,3),EV(3,3),ECOMEG(3,3),EOMEG(3,3),EOMEGD(3,3),DRA(3
     1),DRV(3),DRR(3),Y1S(3),Y2S(3),Y3S(3),Y1SD(3),Y2SD(3),YRPS(3),YRVS(
     23),X1(3),X2(3),X3(3),X4(3),X1D(3),X2D(3),R1(3),R2(3),E(3,3),ER(3,3
     3)
      EQUIVALENCE (EA(1,1),A(391)),(EV(1,1),A(400)),(ECOMEG(1,1),A(409))
     1            (EOMEG(1,1),A(427)),(EOMEGD(1,1),A(418)),
     2            (DRA(1),A(436)),(DRV(1),A(439)),(DRR(1),A(442)),
     3            (Y1S(1),A(311)),(Y2S(1),A(314)),(Y3S(1),A(317)),
     4            (Y1SD(1),A(320)),(Y2SD(1),A(323)),(YRPS(1),A(326)),
     5            (YRVS(1),A(329)),(X1(1),A(007)),(X2(1),A(042)),
     6            (X3(1),A(031)),(X4(1),A(002)),(X1D(1),A(017)),
     7            (X2D(1),A(045)),(E(1,1),A(054))
      IF(MODE)100,300,300
  100 RETURN
  300 CONTINUE
C
C  COMPUTE Y1SDOT
C
      R1(1)=X1D(1)+(-X2D(3)*DRA(2)+X2D(2)*DRA(3))
      R1(2)=X1D(2)+( X2D(3)*DRA(1)-X2D(1)*DRA(3))
      R1(3)=X1D(3)+(-X2D(2)*DRA(1)+X2D(1)*DRA(2))
      R1(1)=R1(1)+(-X2(3)*X1(2)+X2(2)*X1(3))
      R1(2)=R1(2)+( X2(3)*X1(1)-X2(1)*X1(3))
      R1(3)=R1(3)+(-X2(2)*X1(1)+X2(1)*X1(2))
      R2(1)=(-X2(3)*DRA(2)+X2(2)*DRA(3))
      R2(2)=( X2(3)*DRA(1)-X2(1)*DRA(3))
      R2(3)=(-X2(2)*DRA(1)+X2(1)*DRA(2))
      R1(1)=R1(1)+(-X2(3)*R2(2)+X2(2)*R2(3))
      R1(2)=R1(2)+( X2(3)*R2(1)-X2(1)*R2(3))
      R1(3)=R1(3)+(-X2(2)*R2(1)+X2(1)*R2(2))
      DO 1 I=1,3
      Y1SD(I)=0.
      DO 1 J=1,3
    1 Y1SD(I)=Y1SD(I)+EA(I,J)*R1(J)
C
C COMPUTE Y2SDOT
C
      DO 2 I=1,3
      Y2SD(I)=0.
      DO 2 J=1,3
    2 Y2SD(I)=Y2SD(I)+EOMEGD(I,J)*X2D(J)
C
C COMPUTE Y1S
C
      R1(1)=X1(1)+(-X2(3)*DRV(2)+X2(2)*DRV(3))
      R1(2)=X1(2)+( X2(3)*DRV(1)-X2(1)*DRV(3))
      R1(3)=X1(3)+(-X2(2)*DRV(1)+X2(1)*DRV(2))
      DO 3 I=1,3
      Y1S(I)=0.
      DO 3 J=1,3
    3 Y1S(I)=Y1S(I)+EV(I,J)*R1(J)
```

Figure 28. Subroutine SENK Program Listing

```
C COMPUTE Y2S
C
      DO 4 I=1,3
      Y2S(I)=0.
      DO 4 J=1,3
    4 Y2S(I)=Y2S(I)+EOMEG(I,J)*X2(J)
C
C COMPUTE Y3S
C
      DO 5 I=1,3
      Y3S(I)=0.
      DO 5 J=1,3
    5 Y3S(I)=Y3S(I)+ECOMEG(I,J)*X3(J)
C
C COMPUTE YRPS
C
      DO 6 I=1,3
      R1(I)=-DRR(I)
      DO 6 J=1,3
    6 R1(I)=R1(I)-E(I,J)*X4(J)
      XYZ=SQRT(R1(1)*R1(1)+R1(2)*R1(2))
      YRPS(1)=SQRT(XYZ*XYZ+R1(3)*R1(3))
      YRPS(2)=ATAN2(R1(2),R1(1))
      YRPS(3)=ATAN2(R1(3),XYZ)
      STHR=SIN(YRPS(3))
      CTHR=COS(YRPS(3))
      SPSIR=SIN(YRPS(2))
      CPSIR=COS(YRPS(2))
C
C COMPUTE YRVS
C
      ER(1,1)=CTHR*CPSIR
      ER(1,2)=CTHR*SPSIR
      ER(1,3)=-STHR
      ER(2,1)=-SPSIR
      ER(2,2)=CPSIR
      ER(2,3)=0.
      ER(3,1)=STHR*CPSIR
      ER(3,2)=STHR*SPSIR
      ER(3,3)=CTHR
      R1(1)=X1(1)+(-X2(3)*DRR(2)+X2(2)*DRR(3))
      R1(2)=X1(2)+( X2(3)*DRR(1)-X2(1)*DRR(3))
      R1(3)=X1(3)+(-X2(2)*DRR(1)+X2(1)*DRR(2))
      DO 8 I=1,3
      R2(I)=0.
      DO 8 J=1,3
    8 R2(I)=R2(I)+ER(I,J)*R1(J)
      YRVS(1)=-R2(1)
      YRVS(2)=-R2(2)/YRPS(1)
      YRVS(3)=R2(3)/YRPS(1)
      RETURN
      END
```

Figure 28. Subroutine SENK Program Listing (concluded)

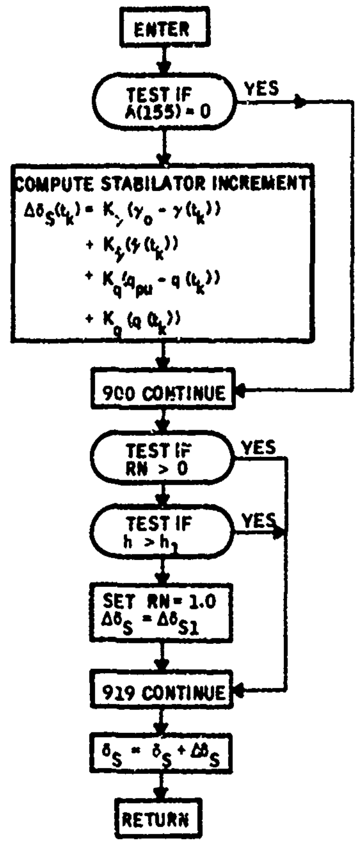


Figure 29. Subroutine PILOT Flow Diagram

```
      SUBROUTINE PILOT(RN)
      COMMON/ADAP/MODE,A(1000)
      IF(A(155).EQ.0.) GOTO 900
      A(122)=A(259)*(A(260)-A(028))
      A(122)=A(122)+A(258)*A(036)
      A(122)=A(122)+A(257)*(A(270)-A(043))
      A(122)=A(122)+A(256)*A(046)
  900 CONTINUE
      IF(RN.GT.0.) GOTO 919
      IF(A(004).GT.A(156)) GOTO 919
      RN=1.
      A(122)=A(157)
  919 CONTINUE
      A(121)=A(121)+A(122)
      RETURN
      END
```

Figure 30. Subroutine PILOT Program Listing

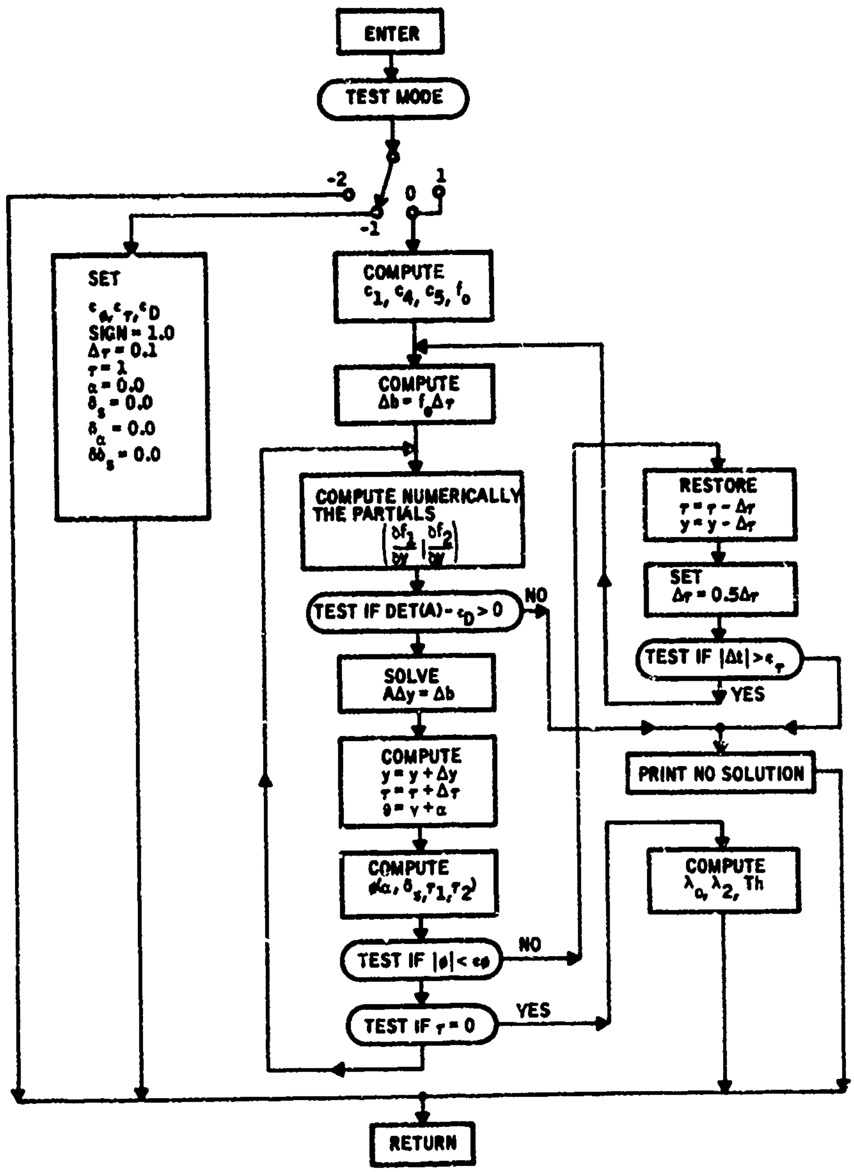


Figure 31. Subroutine NOMK Flow Diagram

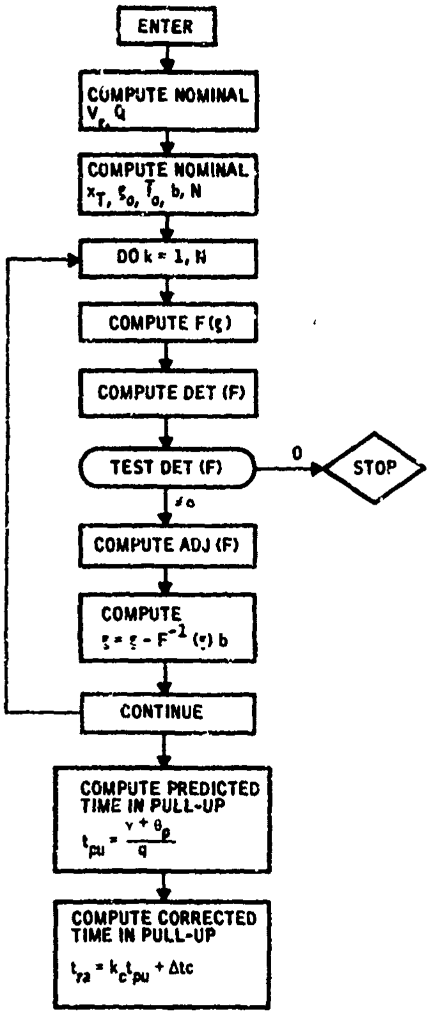


Figure 32. Subroutine RELK Flow Diagram

Subroutine LINK

Subroutine LINK implements the analysis developed in Section VI of Volume I. It generates the linearized equations of motion of the aircraft for six degrees of freedom. It develops the time-varying linear wind filter coefficients by calling subroutine WINDK at the linearization time points. It outputs the linear data for equations of motion, wind filters, and the linearized measurement matrix in the printed form. It also stores these data in a permanent disc file. The subroutine LINK flow diagram is shown in Figure 33 and the program listing in Figure 34. Symbols are listed in Table XVII.

Subroutine SLINK

Subroutine SLINK generates the linear measurement matrix using the nonlinear output from subroutine SENK. Its programming logic is the same as subroutine LINK.

The subroutine SLINK flow diagram is shown in Figure 35 and the program listing in Figure 36.

Subroutine WLINK

Subroutine WLINK generates the linear data for the weapon. Its program logic is the same as subroutine LINK. The linear data is output in print form. The data are also stored in a separate permanent disc file. The subroutine WLINK program listing is shown in Figure 37.

ADAP 1 AUXILIARY SUBROUTINES

Subroutine EXEK

Subroutine EXEK handles the card input for ADAP 1 plus the following bookkeeping functions:

- Keeps track of simulation time.
- Determines integration step size with parameters supplied by the user.
- Prints all data input cards for one run.
- Dumps the A-array contents at the specified time points.
- Stops the simulation.

Subroutine EXEK makes use of the subroutines PRINT, PREAD, and FLOOK. Its flow diagram is shown in Figure 38 and its program listing in Figure 39.

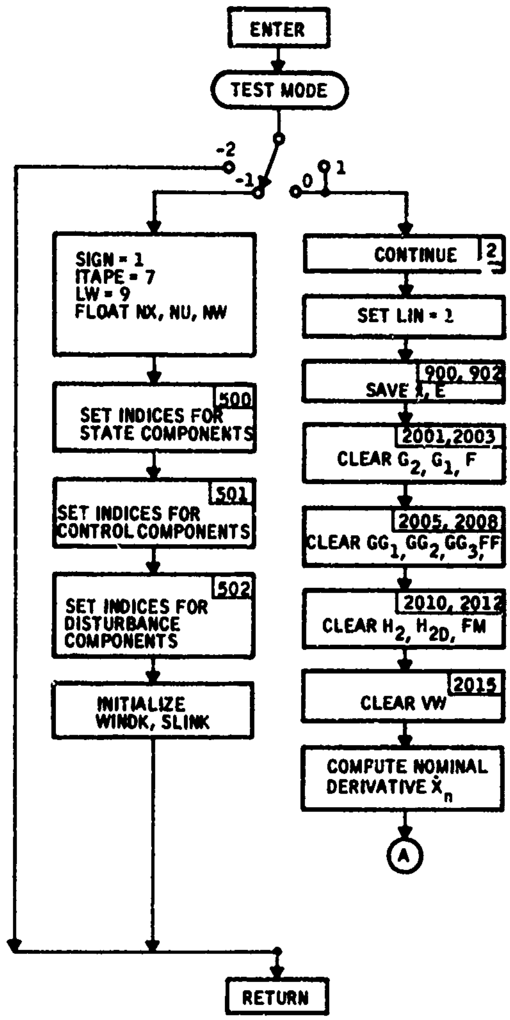


Figure 33. Subroutine LINK Flow Diagram

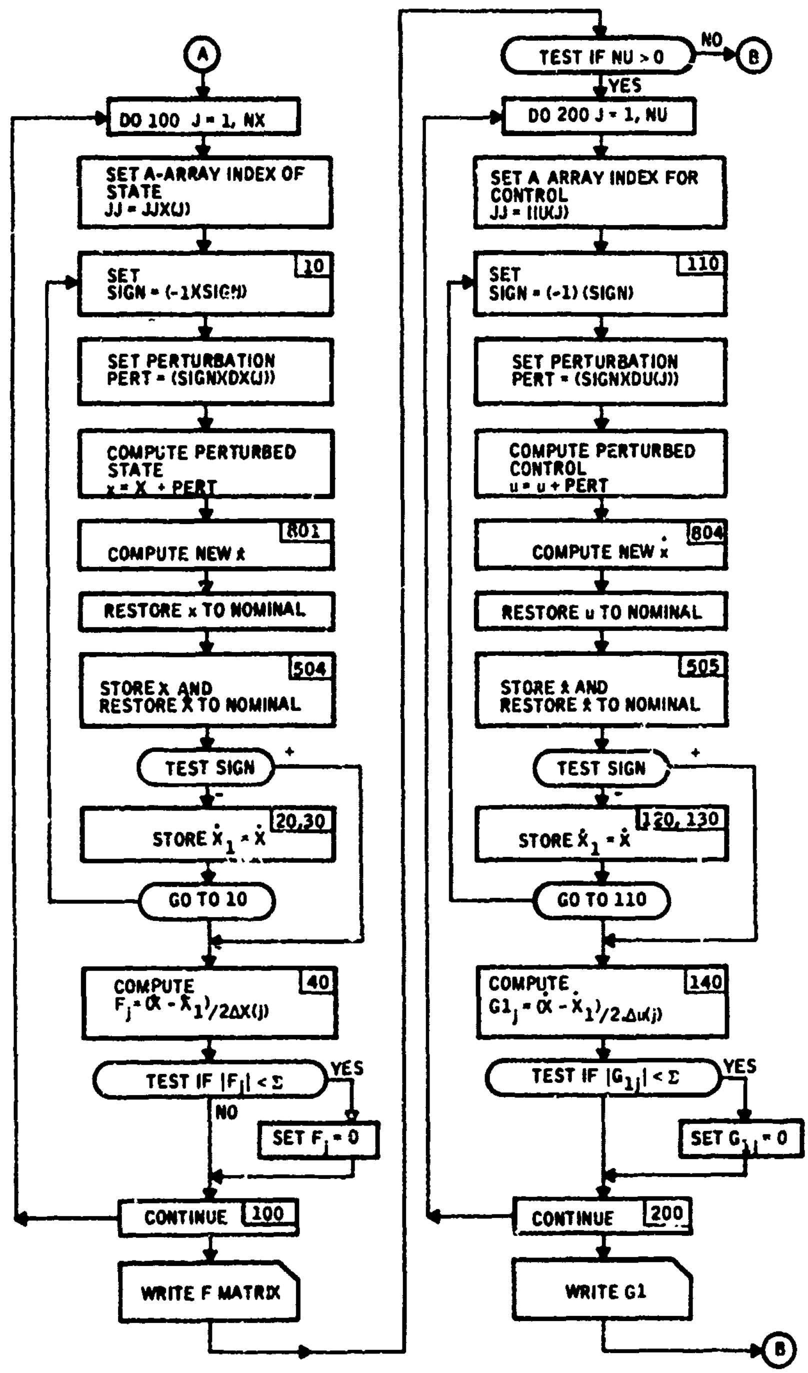


Figure 33. Subroutine LINK Flow Diagram (continued)

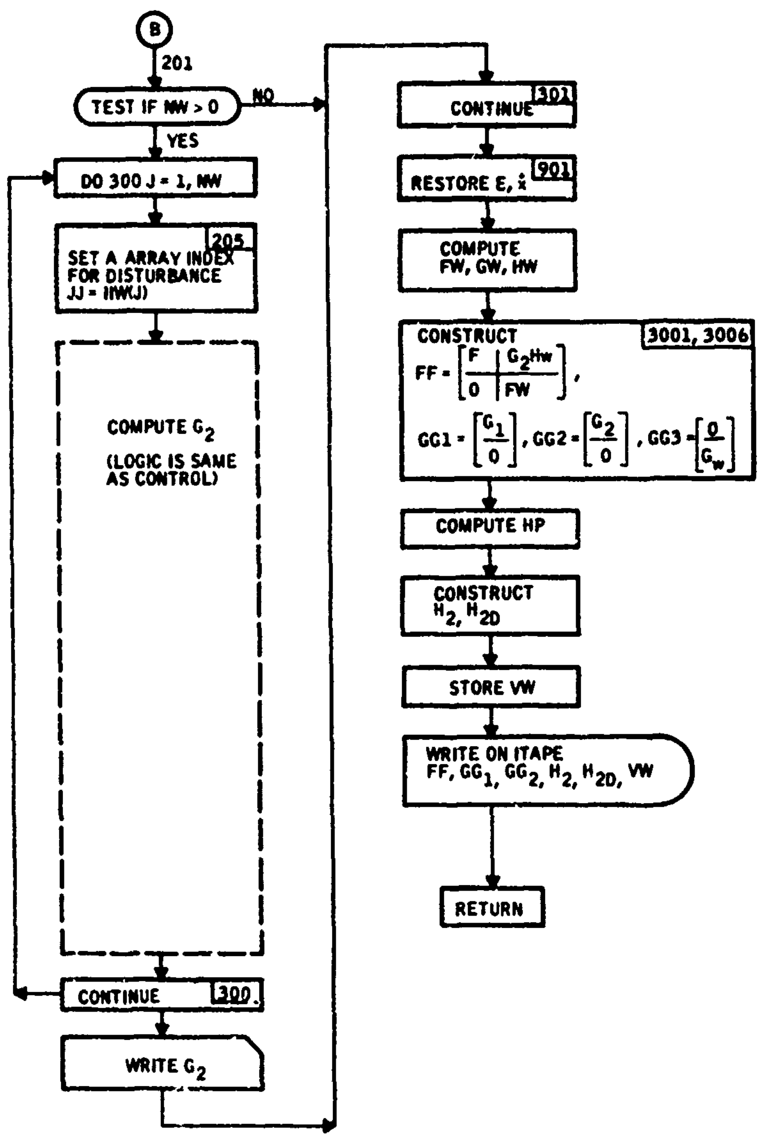


Figure 33. Subroutine LINK Flow Diagram (concluded)

```
      SUBROUTINE LINK
      COMMON/ADAP/MODE,A(1000)
      COMMON IIX(12),IIU(8),IIW(6),IIXD(12)
      COMMON XDOT(12),XDOT1(12),F(12,12),G1(12,8),G2(12,6)
      COMMON    E(9),XS(12),XDOTN(12)
      DIMENSION DX(12),DU(8),DW(6),AI(40)
      DIMENSION FF(20,20),GG1(20,8),GG2(20,6),GG3(20,4),H2(21,12),VW(3)
      DIMENSION FW(8,8),GW(8,4),HW(6,8),H2D(21,6),FM(21,18)
C
C                   **** PARAMETER INPUTS ****
C
      EQUIVALENCE  (DX(1) ,A(201)),(DU(1),A(221)),(DW(1) ,A(231)),
     1             (ANX   ,A(170)),(ANU   ,A(171)),(ANW   ,A(172)),
     2             (AI(1) ,A(901))
      IF(MODE)1,2,2
    1 IF(MODE.LE.-2) RETURN
      SIGN=1.
C
C     SET INDEX ARRAYS
C
      NX=ANX
      NU=ANU
      NW=ANW
      LW=9
      ITAPE=7
      DO 500 I=1,NX
      IIX(I) =AI(I)
      II=I+NX
  500 IIXD(I)=AI(II)
      IB=2*NX+1
      IF=IB+NU-1
      DO 501 I=IB,IF
      J=I-2*NX
  501 IIU(J)=AI(I)
      IB=IF+1
      IE=IB+NW-1
      DO 502 I=IB,IF
      J=I-2*NX-NU
  502 IIW(J)=AI(I)
      CALL WINDK(FW,GW,HW)
      CALL SLINK(FM)
      RETURN
C     PARTIALS W.R.T. STATE
    2 CONTINUE
      LIN=1
C
C SAVE E
C
      DO 902 I=1,NX
      II=IIXD(I)
  902 XS(I)=A(II)
      DO 900 I=1,9
      JK=53+I
  900 E(I)=A(JK)
```

Figure 34. Subroutine LINK Program Listing

```
      DO 2001 I=1,12
      DO 2002 J=1,6
 2002 G2(I,J)=0.
      DO 2003 J=1,8
 2003 G1(I,J)=0.
      DO 2001 J=1,12
 2001 F(I,J)=0.
      DO 2005 I=1,20
      DO 2006 J=1,8
 2006 GG1(I,J)=0.
      DO 2007 J=1,6
 2007 GG2(I,J)=0.
      DO 2008 J=1,4
 2008 GG3(I,J)=0.
      DO 2005 J=1,20
 2005 FF(I,J)=0.
      DO 2010 I=1,21
      DO 2011 J=1,12
 2011 H2(I,J)=0.
      DO 2012 J=1,6
 2012 H2D(I,J)=0.
      DO 2010 J=1,18
 2010 FM(I,J)=0.
      DO 2015 I=1,3
 2015 VM(I)=0.
      CALL AERK
      CALL DYNK(LIN)
      DO 980 I=1,NX
      II=IIXD(I)
  980 XDOTN(I)=A(II)
      DO 100 J=1,NX
      JJ=IIX(J)
   10 SIGN=-1.*SIGN
      PERT= SIGN*DX(J)
      A(JJ)=A(JJ)+PERT
  801 CALL  AERK
      CALL DYNK(LIN)
      A(JJ)=A(JJ)-PERT
      DO 504 I=1,NX
      II=IIXD(I)
      XDOT(I)=A(II)
  504 A(II)=XDOTN(I)
      IF(SIGN)20,20,40
   20 DO 30 I=1,NX
   30 XDOT1(I)=XDOT(I)
      GOTO 10
   40 DO 100 I=1,NX
      F(I,J)=(XDOT(I)-XDOT1(I))/(2.*DX(J))
      IF(ABS(F(I,J)).LT..1E-6) F(I,J)=0.
  100 CONTINUE
      WRITE(LW,700)
      CALL MP(12,12,NX,NX,F,LW)
  700 FORMAT(1H1/7X,10H F MATRIX //)
      IF(NU)201,201,105
```

Figure 34. Subroutine LINK Program Listing (continued)

```
C       PARTIALS W.R.T. STATE
  105 DO 200 J=1,NU
      JJ=IIU(J)
  110 SIGN=-1.*SIGN
      PERT=SIGN*DU(J)
      A(JJ)=A(JJ)+PERT
  804 CALL  AERK
      CALL DYNK(LIN)
      A(JJ)=A(JJ)-PERT
      DO 505 I=1,NX
      II=IIXD(I)
      XDOT(I)=A(II)
  505 A(II)=XDOTN(I)
      IF(SIGN)120,120,140
  120 DO 130 I=1,NX
  130 XDOT1(I)=XDOT(I)
      GOTO 110
  140 DO 200 I=1,NX
      G1(I,J)=(XDOT(I)-XDOT1(I))/(2.*DU(J))
      IF(ABS(G1(I,J)).LT..1E-6) G1(I,J)=0.
  200 CONTINUE
      WRITE(LW,701)
  701 FORMAT(1H1/7X,10H G1 MATRIX//)
      CALL MP(12,8,NX,NU,G1,LW)
  201 IF(NW)301,301,205
C       PARTIALS W.R.T. DISTURBANCE PARAMETERS
  205 DO 300 J=1,NW
      JJ=IIW(J)
  210 SIGN=-1.*SIGN
      PERT=SIGN*DW(J)
      A(JJ)=A(JJ)+PERT
  807 CALL  AERK
      CALL DYNK(LIN)
      A(JJ)=A(JJ)-PERT
      DO 506 I=1,NX
      II=IIXD(I)
      XDOT(I)=A(II)
  506 A(II)=XDOTN(I)
      IF(SIGN)220,220,240
  220 DO 230 I=1,NX
  230 XDOT1(I)=XDOT(I)
      GOTO 210
  240 DO 300 I=1,NX
      G2(I,J)=(XDOT(I)-XDOT1(I))/(2.*DW(J))
      IF(ABS(G2(I,J)).LT..1E-6) G2(I,J)=0.
  300 CONTINUE
      WRITE(LW,702)
  702 FORMAT(1H1/7X,10H G2 MATRIX//)
      CALL MP(12,6,NX,NW,G2,LW)
  301 CONTINUE
C
C
C RESTORE F
C
```

Figure 34. Subroutine LINK Program Listing (continued)

```
      DO 901 I=1,9
      JK=53+I
  901 A(JK)=E(I)
      DO 903 I=1,NX
      II=IIXD(I)
  903 A(II)=XS(I)
      CALL WINDK(FW,GW,HW)
      DO 3001 I=1,12
      DO 3002 J=1,12
 3002 FF(I,J)=F(I,J)
      DO 3001 J=1,6
 3001 GG2(I,J)=G2(I,J)
      DO 3003 I=1,8
      II=I+12
      DO 3003 J=1,8
      JJ=J+12
 3003 FF(II,JJ)=FW(I,J)
      DO 3004 I=1,8
      II=I+12
      DO 3004 J=1,4
 3004 GG3(II,J)=GW(I,J)
      DO 3005 I=1,12
      DO 3005 J=1,8
 3005 GG1(I,J)=G1(I,J)
      DO 3006 I=1,12
      DO 3006 J=1,8
      JJ=J+12
      DO 3006 K=1,NW
 3006 FF(I,JJ)=FF(I,JJ)+G2(I,K)*HW(K,J)
      CALL SLINK(FM)
      DO 3007 I=1,21
      DO 3008 J=1,12
 3008 H2(I,J)=FM(I,J)
      DO 3007 J=1,6
      JJ=J+12
 3007 H2D(I,J)=FM(I,JJ)
      VW(1)=A(298)
      VW(2)=A(299)
      VW(3)=A(300)
C WRITE ON DISK
      WRITE(ITAPE)FF
      WRITE(ITAPE)GG1
      WRITE(ITAPE)GG2
      WRITE(ITAPE)GG3
      WRITE(ITAPE)H2
C     WRITE(ITAPE)H2D
      WRITE(ITAPE)VW
      RETURN
      END
```

Figure 34. Subroutine LINK Program Listing (concluded)

Table XVII. List of Symbols for Subroutine LINK

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
δu_1	DU(1)	221	deg	X		$\delta(\delta_a)$ Perturbations on aileron, stabilator and rudder
δu_2	DU(2)	222	deg	X		$\delta(\delta_s)$
δu_3	DU(3)	223	deg	X		$\delta(\delta_r)$
δu_4	DU(4)	224	deg	X		$\delta(\delta_{sp})$ Perturbations on spoilers, and speed brakes
δu_5	DU(5)	225	deg	X		$\delta(\delta_{sb})$
δu_6	DU(6)	226				
δu_7	DU(7)	227	percent	X		δT_1 Thrust perturbations
δu_8	DU(8)	228	percent	X		δT_2
δw_1	DW(1)	231	ft/sec	X		δu_a Perturbations on wind components
δw_2	DW(2)	232	ft/sec	X		δv_a
δw_3	DW(3)	233	ft/sec	X		δw_a
δx_1	DX(1)	201	ft/sec	X		δu Perturbations on states
δx_2	DX(2)	202	ft/sec	X		δv
δx_3	DX(3)	203	ft/sec	X		δw
δx_4	DX(4)	204	rad/sec	X		δp
δx_5	DX(5)	205	rad/sec	X		δq
δx_6	DX(6)	206	rad/sec	X		δr
δx_7	DX(7)	207	rad	X		$\delta \theta$
δx_8	DX(8)	208	rad	X		$\delta \phi$
δx_9	DX(9)	209	rad	X		$\delta \psi$

Table XVII. List of Symbols for Subroutine LINK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
δx_{10}	DX(10)	210	ft	X		δx_e Perturbations on states
δx_{11}	DX(11)	211	ft	X		δy_e Perturbations on states
δx_{12}	DX(12)	212	ft	X		δz_e Perturbations on states
F	F(I, J)				X	Linear system transition matrix
G_1	G1(I, J)				X	Linear system control input matrix
G_2	G2(I, J)				X	Linear system disturbance input matrix
n_u	NU, ANU	171		X		Number of control inputs
n_w	NW, ANW	172		X		Number of disturbance inputs
n_x	NX, ANX	170		X		Number of state s
	PERT				X	Perturbation
	SIGN				X	Sign of perturbation
u_1	U(1)	123	deg	X		δ_a Aileron, stabilator, and rudder deflections
u_2	U(2)	121	deg	X		δ_s Aileron, stabilator, and rudder deflections
u_3	U(3)	125	deg	X		δ_r Aileron, stabilator, and rudder deflections

Table XVII. List of Symbols for Subroutine LINK (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
u_4	U(4)	645	deg	X		δ_{sp} Spoiler and speed brakes deflections
u_5	U(5)	646	deg	X		δ_{sb}
u_6	U(6)	648				
u_7	U(7)	606	percent	X		T_1
u_8	U(8)	607	percent	X		T_2 Engine thrusts
w_1	W(1)	101	ft/sec	X		u Wind gust components in body axes
w_2	W(3)	102	ft/sec	X		v
w_3	W(3)	103	ft/sec	X		w
x_1	X(1)	7	ft/sec	X		u
x_2	X(2)	8	ft/sec	X		v Linear velocity states
x_3	X(3)	9	ft/sec	X		w
x_4	X(4)	42	rad/sec	X		p
x_5	X(5)	43	rad/sec	X		q Angular velocity states
x_6	X(6)	44	rad/sec	X		r

Table XVII. *List of Symbols for Subroutine LINK* (continued)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
x_7	X(7)	31	rad		X	θ
x_8	X(8)	32	rad		X	ϕ Attitude states
x_9	X(9)	33	rad		X	ψ
x_{10}	X(10)	2	ft		X	x_e
x_{11}	X('1)	3	ft		X	y_e Position states
x_{12}	X(12)	4	ft		X	z_e
$\dot{x}_1$	XDOT(1)	17	ft/sec^2		X	$\dot{u}$
$\dot{x}_2$	XDOT(2)	18	ft/sec^2		X	$\dot{v}$ Linear acceleration components
$\dot{x}_3$	XDOT(3)	19	ft/sec^2		X	$\dot{w}$
$\dot{x}_4$	XDOT(4)	45	rad/sec^2		X	$\dot{p}$
$\dot{x}_5$	XDOT(5)	46	rad/sec^2		X	$\dot{q}$ Angular accelerations
$\dot{x}_6$	XDOT(6)	47	rad/sec^2		X	$\dot{r}$
$\dot{x}_7$	XDOT(7)	39	rad/sec		X	$\dot{\theta}$
$\dot{x}_8$	XDOT(8)	40	rad/sec		X	$\dot{\phi}$ Attitude rates
$\dot{x}_9$	XDOT(9)	41	rad/sec		X	$\dot{\psi}$
$\dot{x}_{10}$	XDOT(10)	13	ft/sec		X	$\dot{x}_e$
$\dot{x}_{11}$	XDOT(11)	14	ft/sec		X	$\dot{y}_e$ Translation rates
$\dot{x}_{12}$	XDOT(12)	15	ft/sec		X	$\dot{z}_e$

Table XVII. List of Symbols for Subroutine LINK (concluded)

Quantity	Mnemonic	A-Array Index	Units	Input	Output	Description
$\dot{x}_{-1}$	XDOT1(1)		ft/sec^2		X	Value of the state derivatives at minus perturbation
$\dot{x}_{-2}$	XDOT1(2)		ft/sec^2		X	
$\dot{x}_{-3}$	XDOT1(3)		ft/sec^2		X	
$\dot{x}_{-4}$	XDOT1(4)		rad/sec^2		X	
$\dot{x}_{-5}$	XDOT1(5)		rad/sec^2		X	
$\dot{x}_{-6}$	XDOT1(6)		rad/sec^2		X	
$\dot{x}_{-7}$	XDOT1(7)		rad/sec		X	
$\dot{x}_{-8}$	XDOT1(8)		rad/sec		X	
$\dot{x}_{-9}$	XDOT1(9)		rad/sec		X	
$\dot{x}_{-10}$	XDOT1(10)		ft/sec		X	
$\dot{x}_{-11}$	XDOT1(11)		ft/sec		X	
$\dot{x}_{-12}$	XDOT1(12)		ft/sec		X	
u(loc)	AI(1)	901		X		State component location of u (see p. 22)

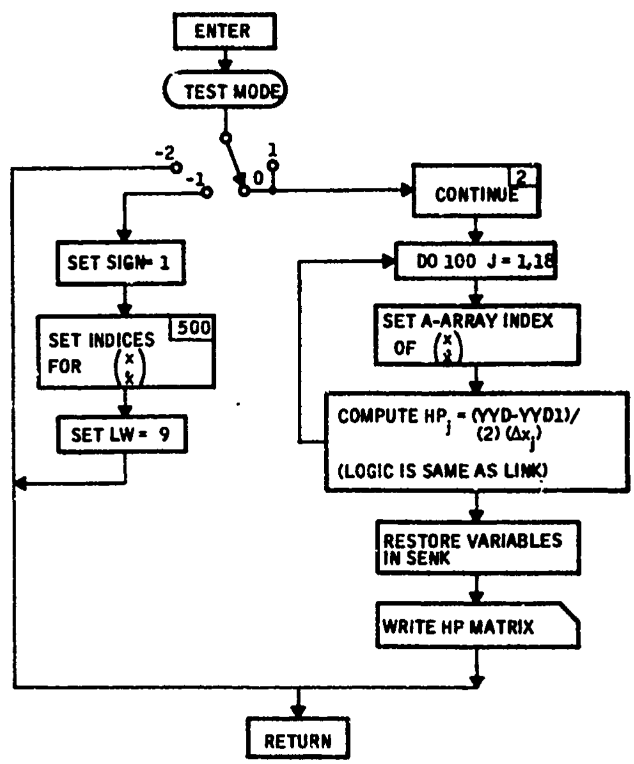


Figure 35. Subroutine SLINK Flow Diagram

```
      SUBROUTINE SLINK(HP)
      COMMON/ADAP/MODE,A(1000)
      DIMENSION HP(21,18),YYD(21),DXXD(18),YYD1(21),IIXXD(18),AI(18)
      EQUIVALENCE (DXXD(1),A(201)),(YYD(1),A(311)),(AI(1),A(901))
      IF (MODE)1,2,2
    1 IF(MODE.LE.-2) RETURN
      SIGN=1.
      DO 500 I=1,18
  500 IIXXD(I)=AI(I)
      LW=9
      RETURN
    2 CONTINUE
      DO 100 J=1,18
      JJ=IIXXD(J)
   10 SIGN=-1.*SIGN
      PERT=SIGN*DXXD(J)
      A(JJ)=A(JJ)+PERT
      CALL SFNK
      A(JJ)=A(JJ)-PERT
      IF(SIGN)20,20,40
   20 DO 504 I=1,21
  504 YYD1(I)=YYD(I)
      GO TO 10
   40 DO 100 I=1,21
      IF(ABS(HP(I,J)).LT..1E-6) HP(I,J)=0.
      HP(I,J)=(YYD(I)-YYD1(I))/(2.*DXXD(J))
  100 CONTINUE
      CALL SFNK
      WRITE (LW,700)
  700 FORMAT(1H1/7X,15H HP      MATRIX//)
      CALL MP(21,18,21,18,HP,LW)
      RETURN
      END
```

Figure 36. Subroutine SLINK Program Listing

```
      SUBROUTINE WLINK
C
      COMMON/ADAP/MODE,A(1000)
      COMMON XDOT(20),XDOT1(20),F(20,20),G1(20,10),G2(20,10)
      COMMON IIX(12),IIU(8),IIW(5),IIXD(12)
      DIMENSION DX(12),DU(8),DW(5),AI(40)
      DIMENSION E(9),XS(20)
      DIMENSION FW(8,8),GW(8,4),HW(6,8),G3(20,4)
C
C                    **** PARAMETER INPUTS ****
C
      EQUIVALENCE  (DX(1) ,A(201)),(DU(1),A(221)),(DW(1) ,A(231)),
     1             (ANX   ,A(170)),(ANU   ,A(171)),(ANW    ,A(172)),
     2             (AI(1) ,A(901))
      IF(MODE)1,2,2
    1 IF(MODE.LE.-2) RETURN
      SIGN=1.
C
C     SET INDEX ARRAYS
C
      NX=ANX
      NU=ANU
      NW=ANW
      LW=9
      ITAPE=6
      DO 500 I=1,NX
      IIX(I) =AI(I)
      II=I+NX
  500 IIXD(I)=AI(II)
      IB=2*NX+1
      IF=IB+NU-1
      DO 501 I=IB,IF
      J=I-2*NX
  501 IIU(J)=AI(I)
      IB=IF+1
      IF=IB+NW-1
      DO 502 I=IB,IF
      J=I-2*NX-NU
  502 IIW(J)=AI(I)
      CALL WINDK(FW,GW,HW)
      RETURN
C     PARTIALS W.R.T. STATE
    2 CONTINUE
      DO 1001 I=1,20
      DO 1002 J=1,20
 1002 F(I,J)=0.
      DO 1001 J=1,10
      G1(I,J)=0.
 1001 G2(I,J)=0.
C
C SAVE E
C
      DO 902 I=1,NX
      II=IIXD(I)
```

Figure 37. Subroutine WLINK Program Listing

```
902 XS(I)=A(II)
    DO 900 I=1,9
    JK=55+I
900 E(I)=A(JK)
    LIN=1
    DO 100 J=1,NX
    JJ=IIX(J)
 10 SIGN=-1.*SIGN
    PERT=SIGN*DX(J)
    A(JJ)=A(JJ)+PERT
801 CALL WAERK
    CALL DYNK(LIN)
    A(JJ)=A(JJ)-PERT
    DO 504 I=1,NX
    II=IIXD(I)
504 XDOT(I)=A(II)
    IF(SIGN)20,20,40
 20 DO 30 I=1,NX
 30 XDOT1(I)=XDOT(I)
    GOTO 10
 40 DO 100 I=1,NX
    F(I,J)=.XDOT(I)-XDOT1(I))/(2.*DX(J))
    IF(ABS(F(I,J)).LT..0000001) F(I,J)=0.
100 CONTINUE
    WRITE(LW,700)
    CALL MP(20,20,NX,NX,F,LW)
700 FORMAT(1H1/7X,10H F MATRIX //)
    IF(NU)201,201,105
C       PARTIALS W.R.T. STATE
105 DO 200 J=1,NU
    JJ=IIU(J)
110 SIGN=-1.*SIGN
    PERT=SIGN*DU(J)
    A(JJ)=A(JJ)+PERT
804 CALL WAERK
    CALL DYNK(LIN)
    A(JJ)=A(JJ)-PERT
    DO 505 I=1,NX
    II=IIXD(I)
505 XDOT(I)=A(II)
    IF(SIGN)120,120,140
120 DO 130 I=1,NX
130 XDOT1(I)=XDOT(I)
    GOTO 110
140 DO 200 I=1,NX
    G1(I,J)=(XDOT(I)-XDOT1(I))/(2.*DU(J))
    IF(ABS(G1(I,J)).LT..0000001) G1(I,J)=0.
200 CONTINUE
    WRITE(LW,701)
701 FORMAT(1H1/7X,10H G1 MATRIX//)
    CALL MP(20,10,NX,NU,G1,LW)
201 IF(NW)301,301,205
C       PARTIALS W.R.T. DISTURBANCE PARAMETERS
205 DO 300 J=1,NW
```

Figure 37. Subroutine WLINK Program Listing (continued)

```
      JJ=IIW(J)
  210 SIGN=-1.*SIGN
      PERT=SIGN*DW(J)
      A(JJ)=A(JJ)+PERT
  807 CALL WAERK
      CALL DYNK(LIN)
      A(JJ)=A(JJ)-PERT
      DO 506 I=1,NX
      II=IIXD(I)
  506 XDOT(I)=A(II)
      IF(SIGN)220,220,240
  220 DO 230 I=1,NX
  230 XDOT1(I)=XDOT(I)
      GOTO 210
  240 DO 300 I=1,NX
      G2(I,J)=(XDOT(I)-XDOT1(I))/(2.*DW(J))
      IF(ABS(G2(I,J)).LT..0000001) G2(I,J)=0.
  300 CONTINUE
      WRITE(LW,702)
  702 FORMAT(1H1/7X,10H G2 MATRIX//)
      CALL MP(20,10,NX,NW,G2,LW)
  301 CONTINUE
C
C
C RESTORE E
C
      DO 901 I=1,9
      JK=53+I
  901 A(JK)=F(I)
      DO 903 I=1,NX
      II=IIXD(I)
  903 A(II)=XS(I)
      CALL WINDK(FW,GW,HW)
      DO 1010 I=1,8
      II=NX+I
      DO 1010 J=1,8
      JJ=NX+J
 1010 F(II,JJ)=FW(I,J)
      DO 1011 I=1,20
      DO 1011 J=1,4
 1011 G3(I,J)=0.
      DO 1012 I=1,8
      II=NX+I
      DO 1012 J=1,4
 1012 G3(II,J)=GW(I,J)
      DO 1013 I=1,NX
      DO 1013 J=1,8
      JJ=NX+J
      DO 1013 K=1,NW
 1013 F(I,JJ)=F(I,JJ)+G2(I,K)*HW(K,J)
      WRITE(ITAPE)F
      WRITE(ITAPE)G3
      RETURN
      END
```

Figure 37. Subroutine WLINK Program Listing (concluded)

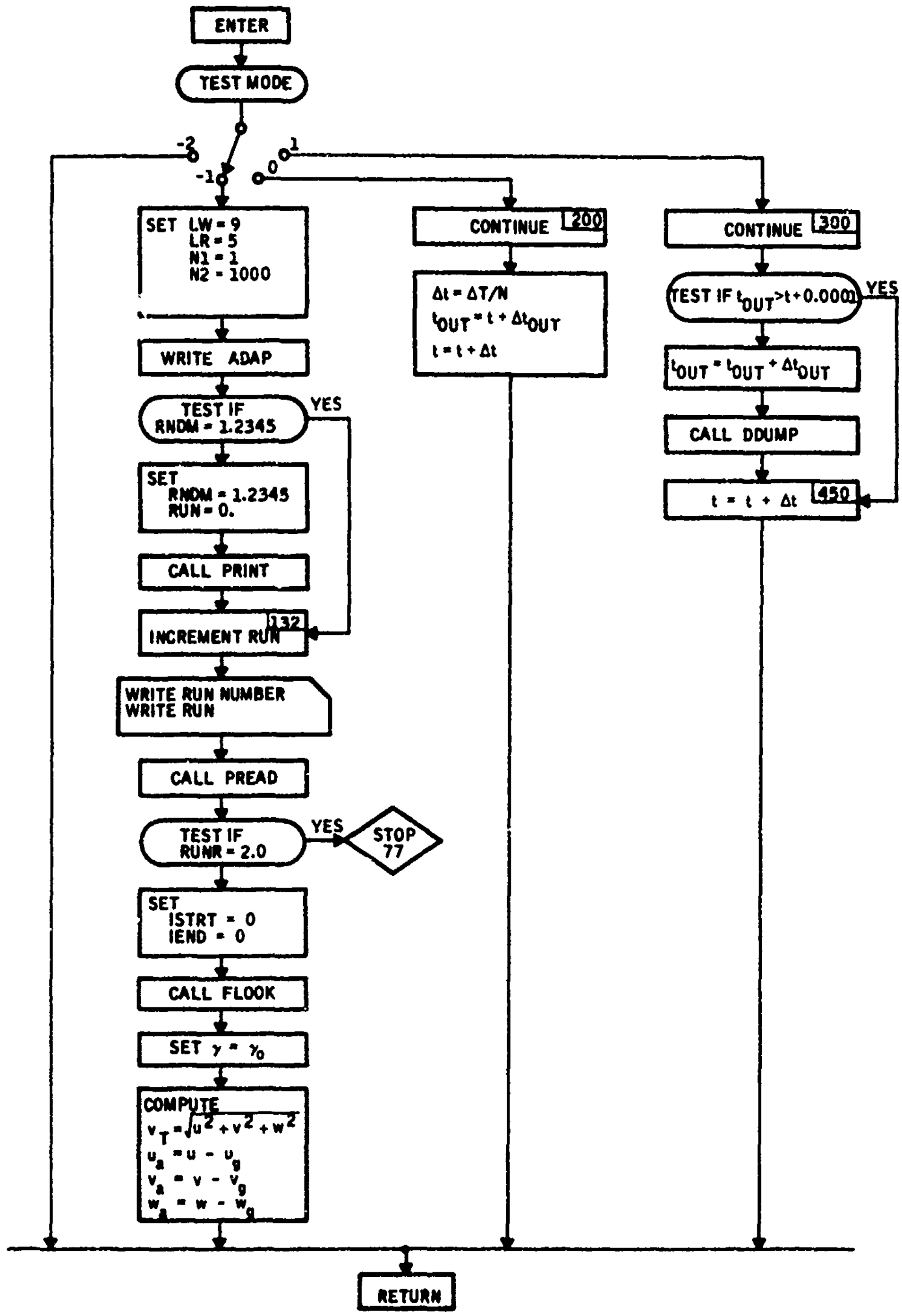


Figure 38. Subroutine EXEK Flow Diagram

```
      SUBROUTINE EXEK
      COMMON/ADAP/MODE,A(1000)
      DIMENSION VST(20),FUN(80)
      EQUIVALENCE (TM     ,A(001)),(DELT   ,A(089)),(DTOUT  ,A(134)),
     1            (GAM    ,A(028)),(GAMN   ,A(260))
      IF(MODE)100,200,300
  100 IF(MODE.LE.-2)RETURN
      LW=9
      LR=5
      A(145)=1.
      A(146)=1000.
      WRITE(LW,131)
  131 FORMAT(1H1,50X,4HADAP//)
      IF(RNDM.EQ.1.2345) GOTO 132
      RNDM=1.2345
      RUN=0.
      CALL PRINT
  132 RUN=RUN+1.
      A(138)=RUN
      WRITE(LW,133)RUN
  133 FORMAT(43X,10HRUN NUMBER,F6.2//)
      CALL PREAD(RUNR,LR,LW)
      IF(RUNR.EQ.2.) STOP 77
      ISTRT=0
      IEND =0
      CALL FLOOK(VST,FUN,ISTRT,IEND)
      GAM=GAMN
      A(725)=SQRT(A(7)*A(7)+A(8)*A(8)+A(9)*A(9))
      A(10)=A(7)-A(101)
      A(11)=A(8)-A(102)
      A(12)=A(9)-A(103)
      RETURN
  200 CONTINUE
C
C COMPUTE DT AND INITIALIZE PERIODIC PRINT
C
      A(89)=A(132)/A(131)
      TOUT=TM+DTOUT
      TM=TM+DELT
      RETURN
  300 CONTINUE
C
C TEST FOR PERIODIC  OUTPUT
C
      IF(TOUT.GT.TM+.0001) GOTO 450
      TOUT=TOUT+DTOUT
      CALL DDUMP(LW)
C
C UPDATE TIME
C
  450 TM=TM+DELT
      RETURN
      END
```

Figure 39. Subroutine EXEK Program Listing

Subroutine FLOOK

Subroutine FLOOK implements the development given in Appendix I of this volume. It reads the card input data for tables in the first call. In subsequent calls, it looks up tables and interpolates to find the value of a function at a given argument set.

The subroutine flow diagram is shown in Figure 40 and the program listing in Figure 41. Symbols are listed in Table XVIII.

Subroutine PREAD

Subroutine PREAD reads the common input cards in ADAP 1 and writes the contents of input cards. It exits either in reading the control card RUN which signifies the end of common card inputs, or in reading the control card STOP which signifies the end of the computer run. It calls subroutine PRINT to read the A-array parameter output specification cards.

The subroutine flow diagram is shown in Figure 42 and the program listing in Figure 43.

Subroutine PRINT

Subroutine PRINT reads A-array output specification cards and prints out the specified A-array contents at the given time points. The subroutine flow diagram is shown in Figure 44 and the program listing in Figure 45.

Subroutine DDUMP

Subroutine DDUMP prints the nonzero elements of the A-array at specified time points. The subroutine flow diagram is shown in Figure 46 and the program listing in Figure 47.

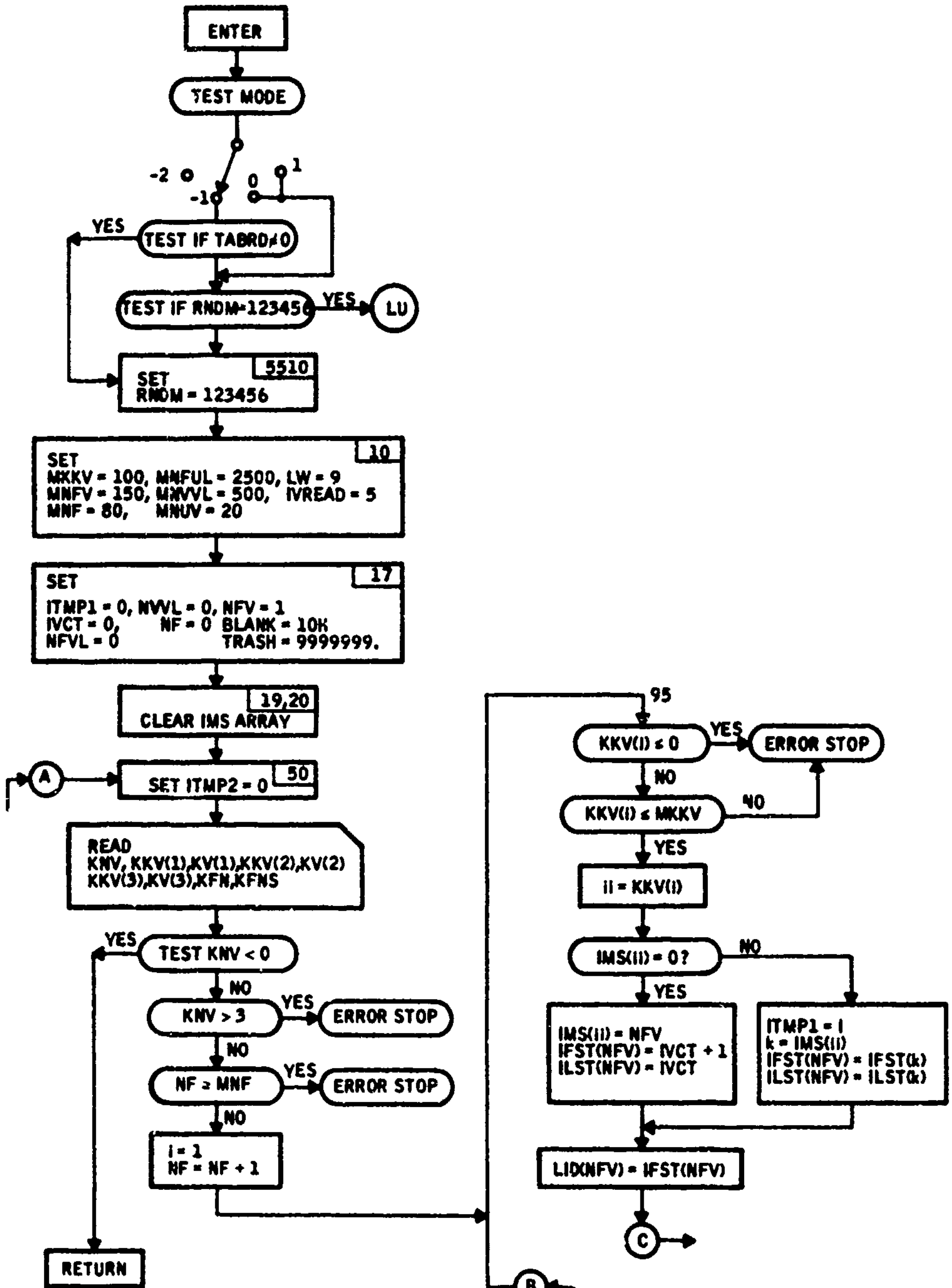


Figure 40. Subroutine FLOOK Flow Diagram

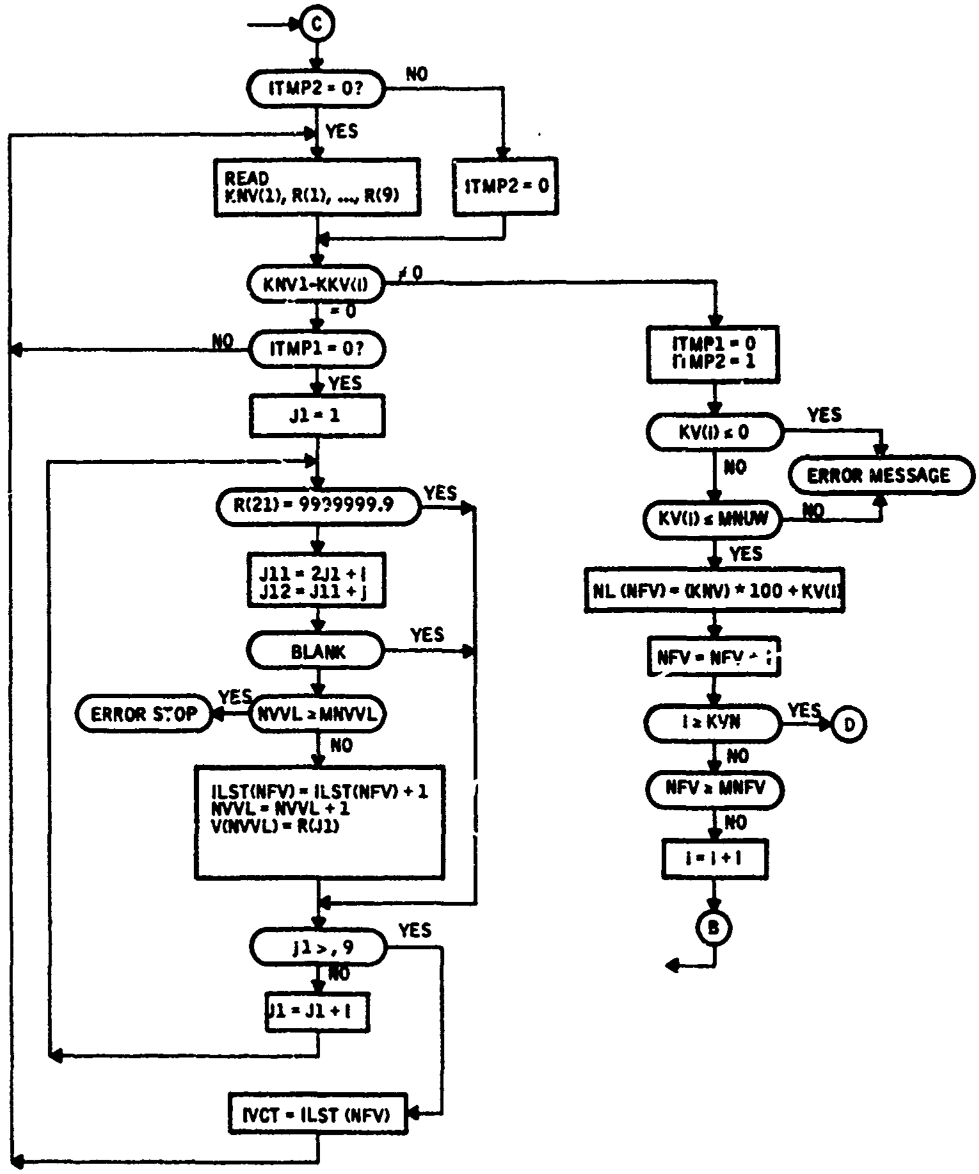


Figure 40. Subroutine FLOOK Flow Diagram (continued)

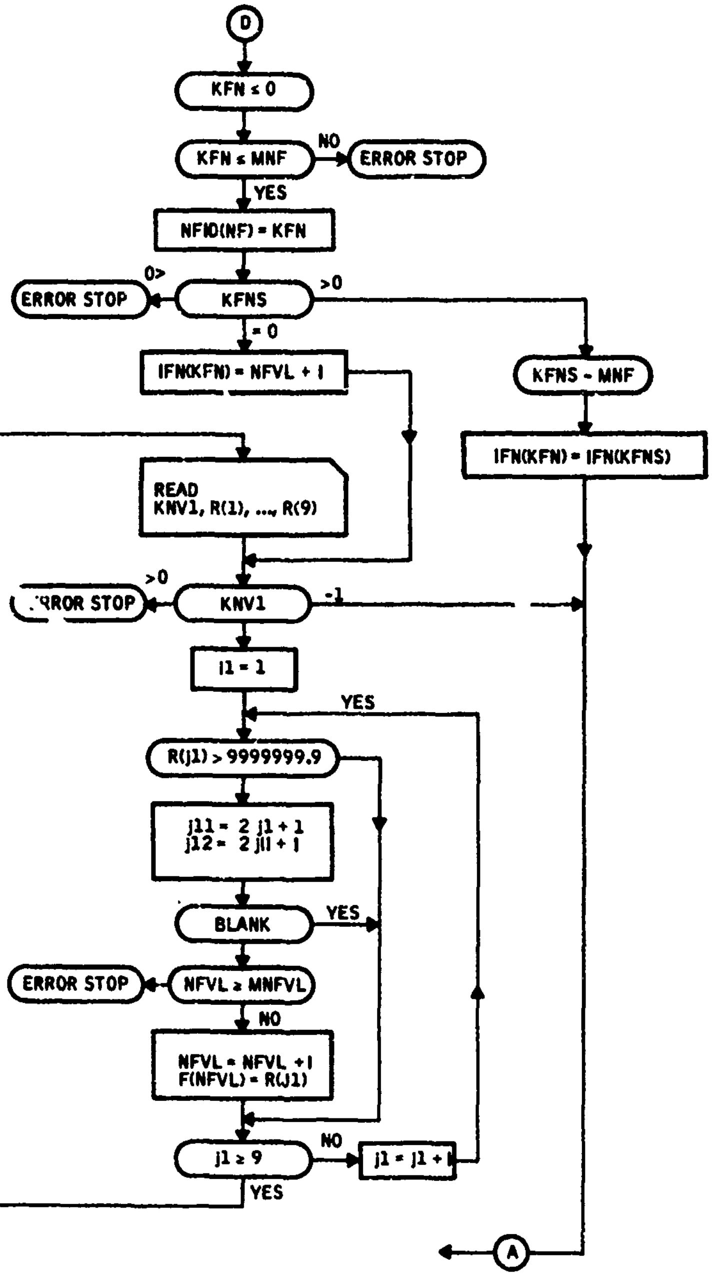


Figure 40. Subroutine FLOOK Flow Diagram (continued)

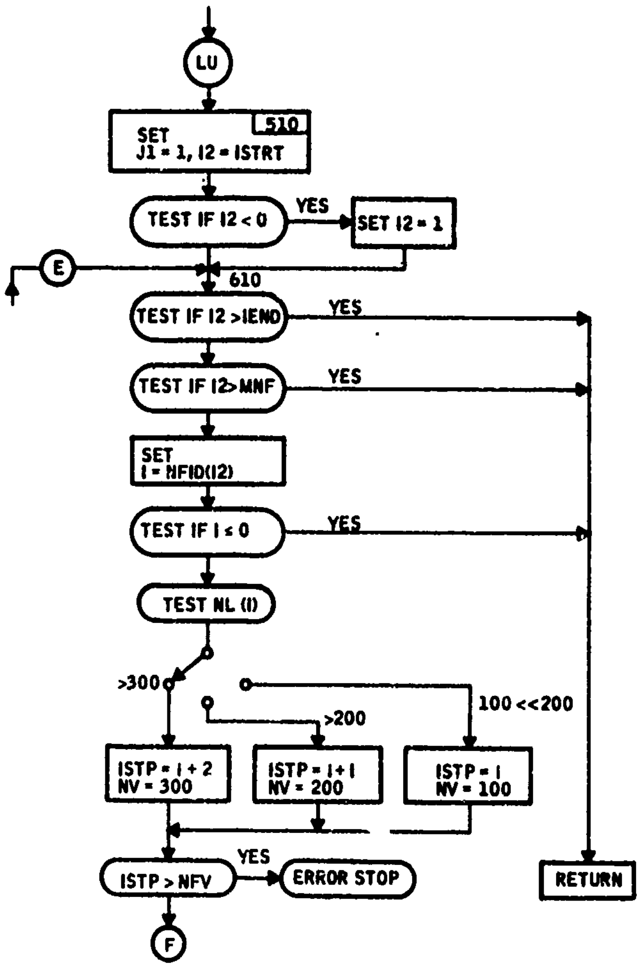


Figure 40. Subroutine FLOOK Flow Diagram (continued)

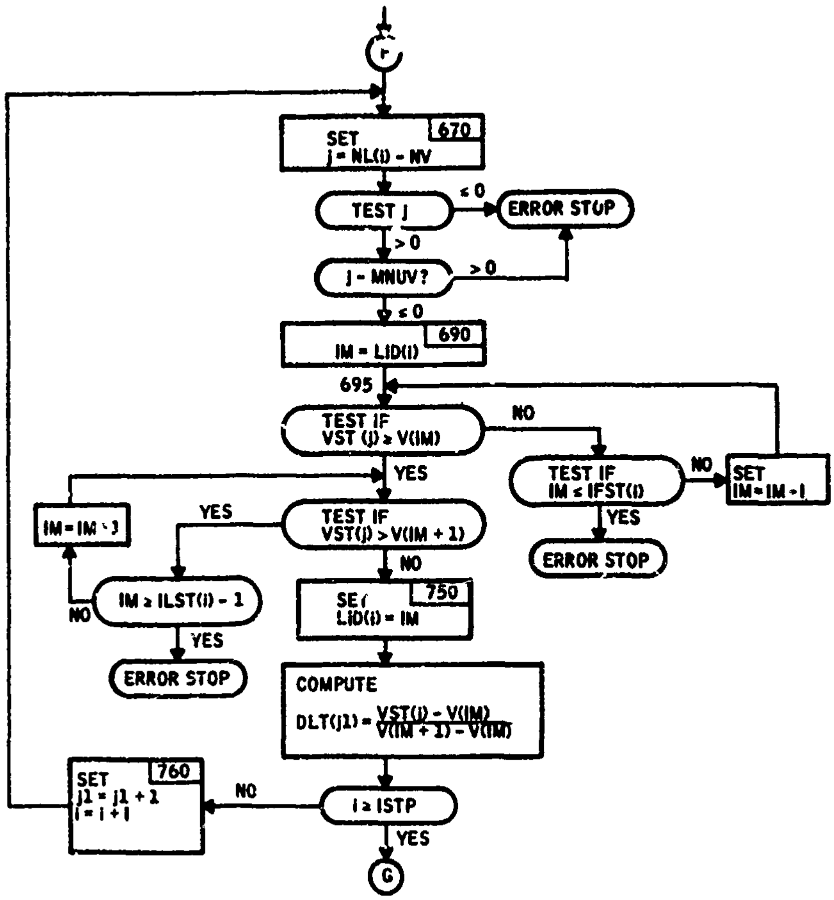


Figure 40. Subroutine FLOOK Flow Diagram (continued)

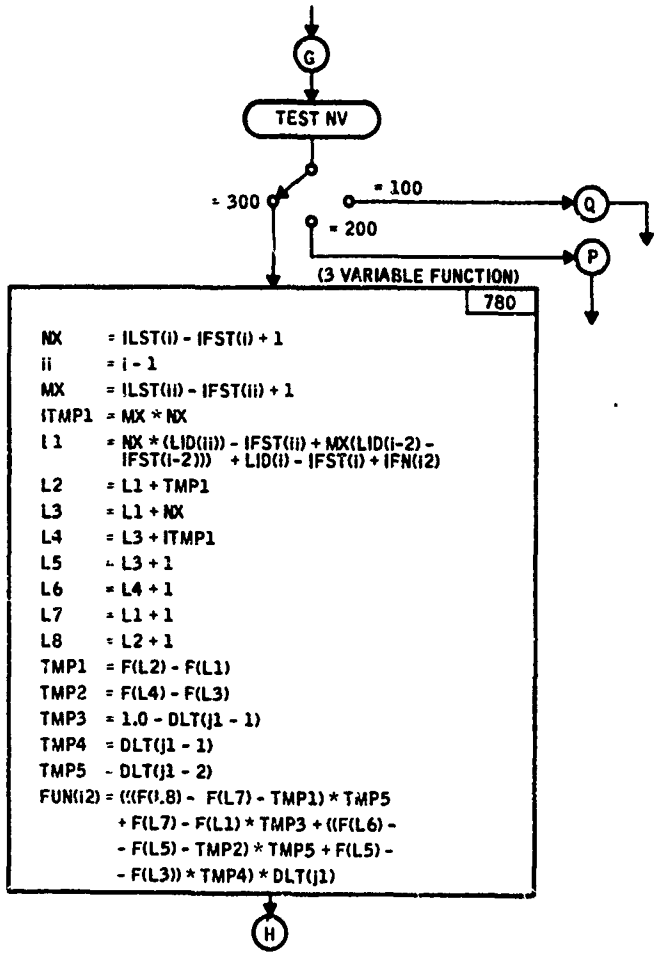


Figure 40. Subroutine FLOOK Flow Diagram (continued)

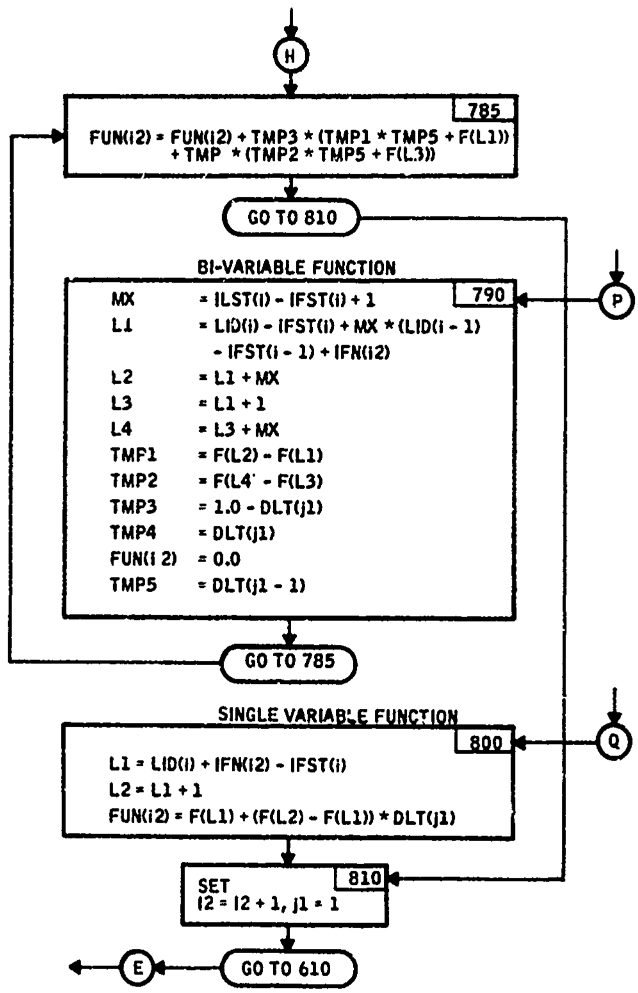


Figure 40. Subroutine FLOOK Flow Diagram (concluded)

```
      SUBROUTINE FLOOK(VST,FUN,ISTRT,IEND)
      COMMON/ADAP/MODE,A(1000)
      EQUIVALENCE(MFN,MNF),(TABRD,A(997))
      DIMENSION IFST(150),ILST(150),IFN(80),F(2500),LID(150),V(500)
      DIMENSION VST(20),NL(150),DLT(3),NFID(80),IMS(100),KKV(3),KV(3)
      DIMENSION R(9),FUN(80),XF(3),RALF(20)
      INTEGER BLANK,RALF
      IF(MODE.EQ.-1.AND.TABRD.NE.0.) GOTO 5510
      IF(RNDM.EQ.123456.) GOTO 510
 5510 RNDM=123456.
C
C     THE ARRAYS SHOULD BE DIMENSIONED AS FOLLOWS
C     DIMENSION A(200),IFST(MNFV),ILST(MNFV),IFN(MNF),F(MNFVL),LID(MNFV),
C               V(MNVVL),VST(MNUV),NL(MNFV),DLT(3),NFID(MNF),IMS(MKKV),
C                KKV(3),KV(3),R(9),FUN(MNF),XF(3),RALF(20)
C     WHERE
   10 MKKV   = 100
      MNFV   = 150
      MNF=80
      MNFVL=2500
      MNVVL=500
      MNUV   = 20
      LW     = 9
      IVREAD = 5
C     AND IF ANY OF THESE VALUES ARE CHANGED, THE DIMENSIONS OF THE
C     APPROPRIATE ARRAYS MUST BE CHANGED.
C
   17 ITMP1 = 0
      IVCT  = 0
      NFVL  = 0
      NFV   = 1
      NVVL  = 0
      NF    = 0
      BLANK=10H
      TRASH = 9999999.0
   19 DO 20 I = 1,MKKV
   20 IMS(I) = 0
   50 ITMP2 = 0
      READ (IVREAD,60) KNV,KKV(1),KV(1),KKV(2),KV(2),KKV(3),KV(3),KFN,
     1KFNS
      WRITE(9,60)      KNV,KKV(1),KV(1),KKV(2),KV(2),KKV(3),KV(3),KFN,
     1KFNS
   60 FORMAT (I5,I4,I2,I4,I2,I4,I2,I4,I4)
      IF (KNV) 500,500,65
   65 IF (KFN.LE.0) GO TO 444
      IF (KFN.GT.MNF) GO TO 444
      NFID(KFN) = NFV
   70 IF ( KNV - 3 ) 80,80,
   80 IF ( NF - MNF ) 90,444,444
   90 I = 1
      NF = NF + 1
   95 IF ( KKV(I) ) 444,444,100
  100 IF ( KKV(I) - MKKV ) 110,110,444
  110 II = KKV(I)
```

Figure 41. Subroutine FLOOK Program Listing

```
      IF ( IMS(II) ) 120,130,120
  120 ITMP1 = 1
      K = IMS(II)
      IFST(NFV) = IFST(K)
      ILST(NFV) = ILST(K)
      GO TO 140
  130 IMS(II) = NFV
      IFST(NFV) = IVCT + 1
      ILST(NFV) = IVCT
  140 LID(NFV) = IFST(NFV)
      IF ( ITMP2 ) 150,160,150
  150 ITMP2 = 0
      GO TO 180
  160 READ(IVREAD,165)(RALF(J1),J1=1,8)
  165 FORMAT(8A10)
      WRITE(9,165)(RALF(J1),J1=1,8)
      DECODE(80,170,RALF)KNV1,(R(J1),J1 = 1,9)
  170 FORMAT (I4,4X,9E8.1)
  180 IF ( KNV1 - KKV(I) ) 260,190,260
  190 IF ( ITMP1 ) 160,200,160
  200 J1 = 1
205   IF(R(J1).GT.TRASH) GO TO 230
      IF(J1.LE.5) GOTO 46
      JJ=J1-1
      J11=J1
      GOTO 47
   46 JJ=J1
      J11=J1+1
   47 CONTINUE
      GOTO(30,31,32,33,34,30,31,32,33),J1
   30 ENCODE(8,40,NAME)RALF(JJ),RALF(J11)
   40 FORMAT(R2,A6)
      GOTO 49
   31 ENCODE(8,41,NAME)RALF(JJ),RALF(J11)
   41 FORMAT(R4,A4)
      GOTO 49
   32 ENCODE(8,42,NAME)RALF(JJ),RALF(J11)
   42 FORMAT(R6,A2)
      GOTO 49
   33 ENCODE(8,43,NAME)RALF(JJ)
   43 FORMAT(R8)
      GOTO 49
   34 ENCODE(8,44,NAME)RALF(JJ)
   44 FORMAT(A8)
   49 CONTINUE
      IF(NAME.EQ.BLANK) GOTO 230
  210 IF (NVVL - MNVVL) 220,444,444
  220 ILST(NFV) = ILST(NFV) + 1
      NVVL = NVVL + 1
      V(NVVL) = R(J1)
  230 IF ( J1 - 9 ) 240,250,250
  240 J1 = J1 + 1
      GO TO 205
  250 IVCT = ILST(NFV)
```

Figure 41. Subroutine FLOOK Program Listing (continued)

```
      GO TO 160
  260 ITMP2 = 1
      ITMP1 = 0
      IF ( KV(I) ) 444,444,270
  270 IF ( KV(I) - MNUV ) 280,280,444
  280 NL(NFV) = KNV*100 + KV(I)
      NFV = NFV + 1
      IF ( I - KNV ) 290,310,310
  290 IF ( NFV - MNFV ) 300,444,444
  300 I = I + 1
      GO TO 95
  310 IF (KFNS) 444,335,336
  335 IFN(KFN) = NFVL + 1
      GO TO 350
  336 IF ( KFNS - MNF ) 337,337,444
  337 IFN(KFN) = IFN(KFNS)
      GO TO 50
  340 READ(IVREAD,165)(RALF(J1),J1=1,8)
      WRITE(9,165)(RALF(J1),J1=1,8)
      DECODE(80,170,RALF)KNV1,(R(J1),J1 = 1,9)
  350 IF ( KNV1 ) 50,360,444
  360 J1 = 1
365   IF(R(J1).GT.TRASH) GO TO 385
      IF(J1.LE.5) GOTO 26
      JJ=J1-1
      J11=J1
      GOTO 27
   26 JJ=J1
      J11=J1+1
   27 CONTINUE
      GOTO(21,22,23,24,25,21,22,23,24),J1
   21 ENCODE(8,'^,NAME)RALF(JJ),RALF(J11)
      GOTO 29
   22 ENCODE(8,    ,AME)RALF(JJ),RALF(J11)
      GOTO 29
   23 ENCODE(8,42,NAME)RALF(JJ),RALF(J11)
      GOTO 29
   24 ENCODE(8,43,NAME)RALF(JJ)
      GOTO 29
   25 ENCODE(8,44,NAME)RALF(JJ)
   29 CONTINUE
      IF(NAME.EQ.BLANK) GOTO 385
  370 IF ( NFVL - MNFVL ) 380,444,444
  380 NFVL = NFVL + 1
      F(NFVL) = R(J1)
  385 IF ( J1 - 9 ) 390,340,340
  390 J1 = J1 + 1
      GO TO 365
  444 WRITE (LW,445) KFN,KNV,KV(1),KV(2),KV(3),KKV(1),KKV(2),KKV(3),
     1              NVVL,MNVVL,NFV,MNFV,NFVL,MNFVL,NF,MNF,KNV1,KKV(1),
     2              KKV(2),KKV(3)
  445 FORMAT (26H FUNCTION TABLE DATA ERROR/18H FUNCTION NUMBER =I5/
     1         22H NUMBER OF VARIABLES =I5/5H V1 =I3 /5H V2 =I3/5H V3 =I3
     2         /7H SET1 =I4/7H SET2 =I4/7H SET3 =I4/7H NVVL =I4,3H---I4/
```

Figure 41. Subroutine FLOOK Program Listing (continued)

```
   3         7H NFV  =I4,3H---I4/7H NFVL =I4,3H---I4/7H NF   =I4,3H---
   4         I4/7H KNV1 =I4,4H---(I2,1H,I2,1H,I2,1H) )
      STOP
  500 RETURN
  510 J1 = 1
      I2 = ISTRT
      IF (I2.LE.0) I2 = 1
  610 IF (I2.GT.IEND.OR.I2.GT.MFN) GO TO 830
      I = NFID(I2)
      IF (I.LE.0) GO TO 810
      IF ( NL(I) - 300 ) 630,2222,620
  620 ISTP = I + 2
      NV - 300
      GO TO 660
  630 IF ( NL(I) - 200 ) 650,2222,640
  640 ISTP = I + 1
      NV = 200
      GO TO 660
  650 ISTP = I
      NV = 100
  660 IF ( ISTP - NFV ) 670,670,2222
  670 J = NL(I) - NV
      IF ( J ) 2222,2222,680
  680 IF ( J - MNUV ) 690,690,2222
  690 IM = LID(I)
  695 IF (VST(J) - V(IM))700,720,720
  700 IF (IM - IFST(I)) 2222,715,710
  710 IM = IM - 1
      GO TO 695
  715 LID(I) = IM
      DLT(J1) = 0.0
      GO TO 755
  720 IF (VST(J) - V(IM + 1))750,750,730
  730 IF (IM - ILST(I) + 1) 740,745,2222
  740 IM = IM + 1
      GO TO 720
  745 LID(I) = IM
      DLT(J1) = 1.0
      GO TO 755
  750 LID(I) = IM
      DLT(J1) = (VST(J) - V(IM))/(V(IM+1) - V(IM))
  755 IF (I - ISTP) 760,770,770
  760 J1 = J1 + 1
      I = I + 1
      GO TO 670
  770 IF (NV - 200) 800,790,780
  780 NX = ILST(I) - IFST(I) + 1
      II = I - 1
      MX = ILST(II) - IFST(II) + 1
      ITMP1 = MX*NX
      L1 = NX*(LID(II) - IFST(II) + MX*(LID(I-2) - IFST(I-2))) +
     1 LID(I) - IFST(I) + IFN(I2)
      L2 = L1 + ITMP1
      L3 = L1 + NX
```

Figure 41. Subroutine FLOOK Program Listing (continued)

```
      L4 = L3 + ITMP1
      L5 = L3 + 1
      L6 = L4 + 1
      L7 = L1 + 1
      L8 = L2 + 1
      TMP1 = F(L2) - F(L1)
      TMP2 = F(L4) - F(L3)
      TMP3 = 1.0 - DLT(J1 - 1)
      TMP4 = DLT(J1 - 1)
      TMP5 = DLT(J1 - 2)
      FUN(I2) = (((F(L8) - F(L7) - TMP1)*TMP5 + F(L7) - F(L1))*TMP3 +
     1          ((F(L6) - F(L5) - TMP2)*TMP5 + F(L5) - F(L3))*TMP4)*
     2          DLT(J1)
  785 FUN(I2) = FUN(I2) + TMP3*(TMP1*TMP5 + F(L1)) +
     1          TMP4*(TMP2*TMP5 + F(L3))
      GO TO 810
  790 MX = ILST(I) - IFST(I) + 1
      L1 = LID(I) - IFST(I) + MX*(LID(I-1) - IFST(I-1)) + IFN(I2)
      L2 = L1 + MX
      L3 = L1 + 1
      L4 = L3 + MX
      TMP1 = F(L2) - F(L1)
      TMP2 = F(L4) - F(L3)
      TMP3 = 1.0 - DLT(J1)
      TMP4 = DLT(J1)
      FUN(I2) = 0.0
      TMP5 = DLT(J1 - 1)
      GO TO 785
  800 L1 = LID(I) - IFST(I) + IFN(I2)
      L2 = L1 + 1
      FUN(I2) = F(L1) + (F(L2) - F(L1))*DLT(J1)
  810 I2 = I2 + 1
      J1 = 1
      GO TO 610
 830  RETURN
 2222 WRITE (9,2223) NFID(K),J,VST(J)
 2223 FORMAT (20H TABLE LOOK-UP ERROR/17H FUNCTION NUMBER I3/
     1        17H VARIABLE NUMBER I3,1H=F11.4)
      INIT = -2
      RETURN
      END
```

Figure 41. Subroutine FLOOK Program Listing (concluded)

Table XVIII. List of Symbols for Subroutine FLOOK

Mnemonic	Description
1TMP1	Switch used to ignore a set of variable values in the input if they have been read previously
1TMP2	Switch used to determine when a variable value card should be read
KNV	Number of variables
KKV(1) KKV(2) KKV(3)	Set numbers associated with the first, second, and third variables, respectively
KV(1) KV(2) KV(3)	First, second, and third variables, respectively.
KFN	Function number
KFNS	Number of the previously read function table with identical function table data.
MNUV	Maximum number of variables
NV	Number of variables
NFV	Reading sequence number
F	Function table array
FUN	Current value of each function after interpolation
LID( )	Starting location for variable value search
NFV	Total number of variables specified, where $f_1(\alpha)$, $f_2(\alpha)$, and $f_3(\alpha)$ count as three variables
NFVL	Total number of function table values that were read
NVVL	Total number or variable values read
NUV	Number of unique variables specified; such as α, β, M, δ_e, p, q, ..., etc.

Table XVIII. List of Symbols for Subroutine FLOOK (concluded)

Mnemonic	Description
F	Function table array
IFN	Beginning location in array for each table of function values
IFST	Array for first location of each variable value set
ILST	Array for last location of each variable value set
IMS	Keeps track of which variable value sets have been read and where they are stored
LID	Equivalent to IFST (see subroutine FLOOK)
NFID	Function numbers, in the order which the functions were read
NL	Number of variables in each function and their identification
KNV1	Set number
R(1) R(2) R(3) R(4) R(5) R(6) R(7) R(8) R(9)	Variable values or function values
V(1) V(2) V(3)	Variable values

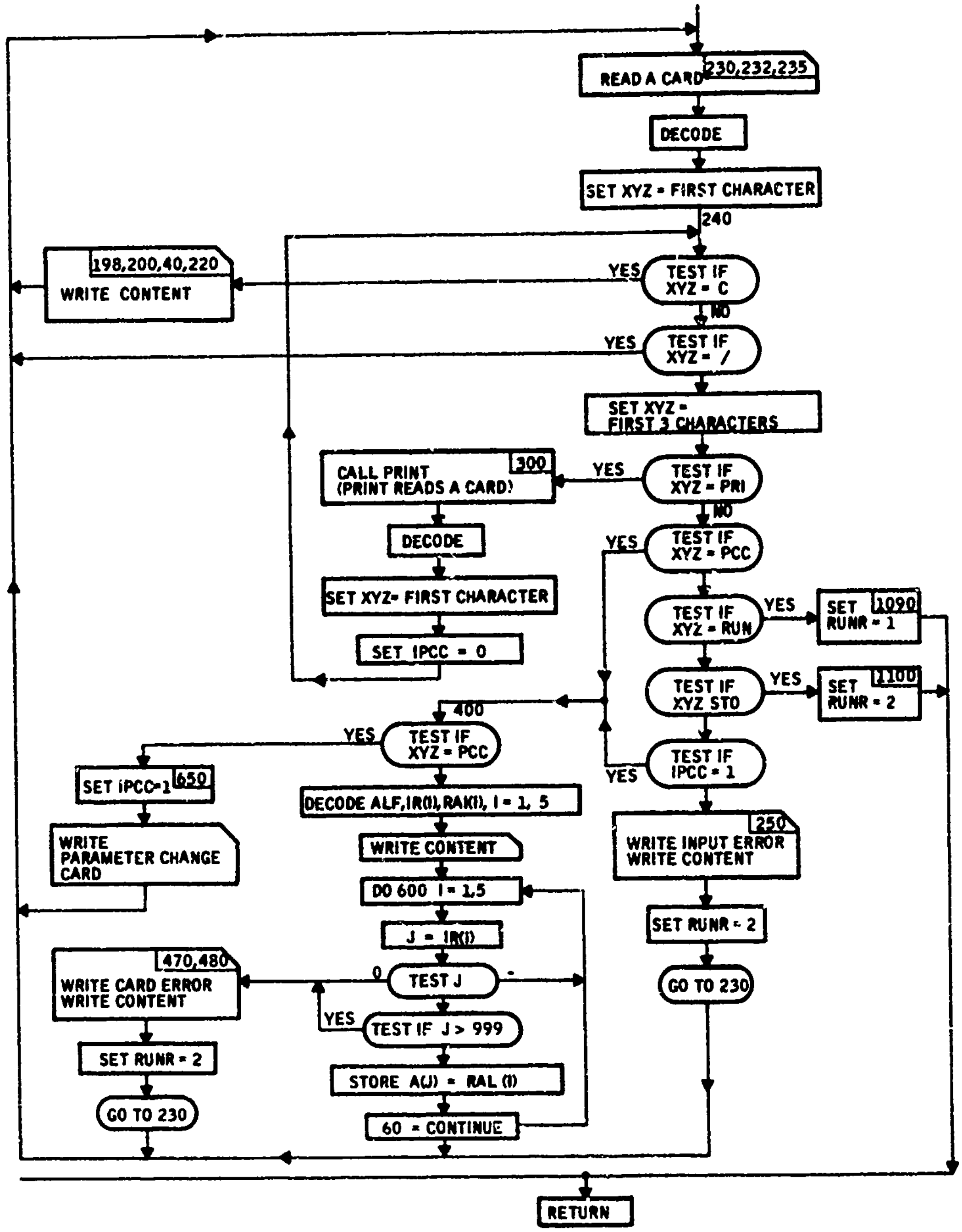


Figure 42. Subroutine PREAD Flow Diagram

```
      SUBROUTINE PREAD(RUNR,LR,LW)
      COMMON/ADAP/MODE,A(1000)
      DIMENSION IR(20),RAL(20)
      COMMON ALF(20)
      INTEGER ALF,XYZ
      IPCC=0
      RUNR=0.
C
C     ****    READ CARD INPUT    ****
C
  230 READ(LR,232)(ALF(I),I=1,8)
  232 FORMAT(8A10)
      DECODE(80,235,ALF)XYZ
  235 FORMAT(A1)
  240 IF(XYZ.EQ.1HC) GOTO 198
      IF(XYZ.EQ.1H/) GOTO 230
      DECODE(80,250,ALF)XYZ
  250 FORMAT(A3)
      IF(XYZ.EQ.3HPRI) GOTO 300
C  PRINT SPECIFICATION CARD
      IF(XYZ.EQ.3HPCC) GOTO 400
C  PARAMETER CHANGE CARD
      IF(XYZ.EQ.3HRUN) GOTO 1090
C  RUN CARD
      IF(XYZ.EQ.3HSTO) GOTO 1100
C  STOP CARD
      IF(IPCC.EQ.1)    GOTO 400
      WRITE(LW,270)(ALF(I),I=1,8)
  270 FORMAT(12H INPUT ERROR/8A10)
  198 DECODE(80,200,ALF)(IR(I),I=1,8)
  200 FORMAT(8A10)
  210 WRITE(LW,220)(IR(I),I=1,8)
  220 FORMAT(20X,8A10)
      GOTO 230
C     ****    READ PRINT SPEC CARDS ****
  300 CALL PRINT
      DECODE(80,235,ALF)XYZ
      IPCC=0
      GOTO 240
  400 IF(XYZ.EQ.3HPCC) GOTO 650
      DECODE(80,420,ALF)(IR(I),RAL(I),I=1,5)
  420 FORMAT(5(I4,E11.4))
  445 WRITE(LW,450)(IR(I),RAL(I),I=1,5)
  450 FORMAT(5(I5,E15.7))
      DO 600 I=1,5
      J=IR(I)
      IF(J)470,600,500
  500 IF( J.GT.999) GOTO 470
      A(J)=RAL(I)
  600 CONTINUE
      GOTO 230
  650 IPCC=1
      WRITE(LW,670)
  670 FORMAT(/23H PARAMETER CHANGE CARDS)
```

Figure 43. Subroutine PREAD Program Listing

```
      GOTO 230
 470  WRITE(LW,480)(IR(K),RAL(K),K=1,5)
 480  FORMAT(11H CARD ERROR/5(I5,E15.7)/)
      MODE=-2
      RETURN
1090  RUNR=1.
      RETURN
1100  RUNR=2.
      RETURN
      END
```

Figure 43. Subroutine PREAD Program Listing (concluded)

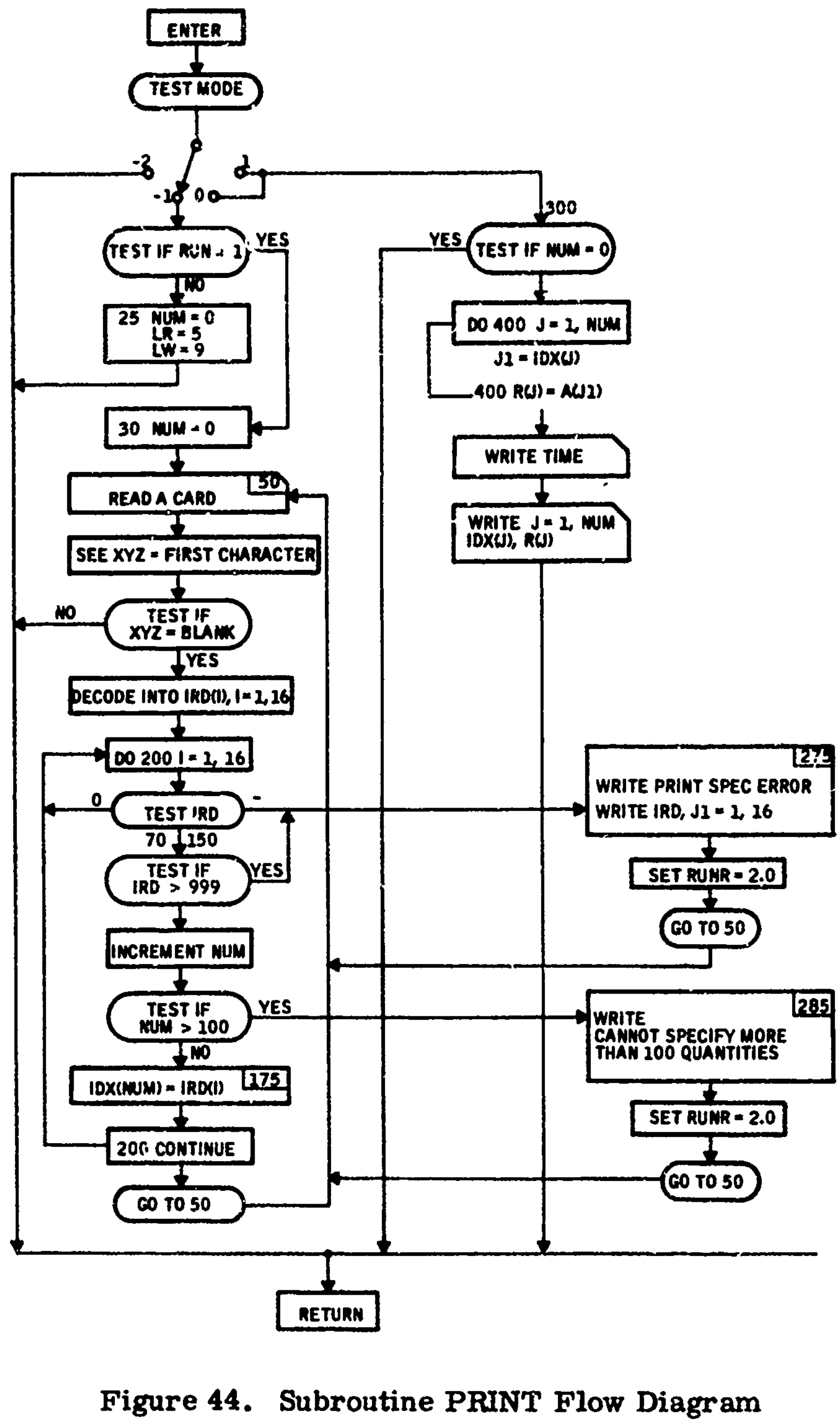


Figure 44. Subroutine PRINT Flow Diagram

```
      SUBROUTINE PRINT
      DIMENSION IRD(16),IDX(100),R(100)
      COMMON/ADAP/MODE,A(1000)
      COMMON ALF(20)
      INTEGER XYZ,ALF
      IF(MODE)20,300,300
   20 IF(A(138)-1.)25,30,30
   25 NUM=0
      LR=5
      LW=9
      RETURN
   30 NUM=0
   50 READ(LR,60)(ALF(I),I=1,8)
   60 FORMAT(8A10)
      DECODE(80,80,ALF)XYZ
   80 FORMAT(A1)
      IF(XYZ.NE.1H ) RETURN
      DECODE(80,100,ALF)(IRD(J1),J1=1,16)
  100 FORMAT(16I5)
  140 DO 200 I=1,16
      IF(IRD(I))275,200,150
  150 IF(IRD(I)-999)160,160,275
  160 NUM=NUM+1
      IF(NUM-100)175,175,285
  175 IDX(NUM)=IRD(I)
  200 CONTINUE
      GOTO 50
  275 WRITE(LW,280)(IRD(J1),J1=1,16)
  280 FORMAT(//17H PRINT SPEC ERROR/16I5)
      STOP
  285 WRITE(LW,290)
  290 FORMAT(//47H CANNOT SPECIFY MORE THAN 100 OUTPUT QUANTITIES)
      STOP
  300 IF(NUM.EQ.0) RETURN
      DO 400 J=1,NUM
      J1=IDX(J)
  400 R(J)=A(J1)
      WRITE(LW,500)A(1)
  500 FORMAT(1H1/7H TIME =F8.3)
      WRITE(LW,600)(IDX(J),R(J),J=1,NUM)
  600 FORMAT(7(I4,E12.4))
      RETURN
      END
```

Figure 45. Subroutine PRINT Program Listing

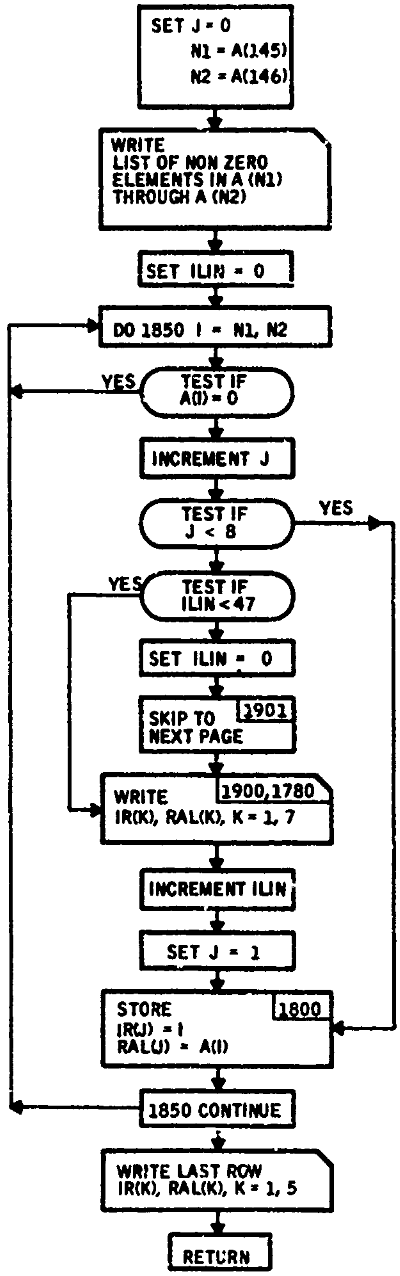


Figure 46. Subroutine DDUMP Flow Diagram

```
      SUBROUTINEDDUMP(LW)
      COMMON/ADAP/MODE,A(1000)
      DIMENSION IR(7),RAL(7)
      J=0
      N1=A(145)
      N2=A(146)
      WRITE(LW,1750)N1,N2
 1750 FORMAT(1H1//32H LIST OF NON-ZERO ELEMENTS IN A(I4,11H), THRU, A(I4
     1,1H)/)
      ILIN=0
      DO 1350 I=N1,N2
      IF(A(I).EQ.0.) GOTO 1850
      J=J+1
      IF(J.LT.8) GOTO 1800
      IF(ILIN.LT.47) GOTO 1900
      ILIN=0
      WRITE(LW,1901)
 1901 FORMAT(1H1//)
 1900 WRITE(LW,1780)(IR(K),RAL(K),K=1,7)
 1780 FORMAT(7(I4,E12.4))
      ILIN=ILIN+1
      J=1
 1800 IR(J)=I
      RAL(J)=A(I)
 1850 CONTINUE
      WRITE(LW,1780)(IR(K),RAL(K),K=1,J)
      RETURN
      END
```

Figure 47. Subroutine DDUMP Program Listing

SECTION IV

ADAP 2 (DISCOP) -- NONSTATIONARY OPTIMIZATION PROGRAM

ADAP 2 is a program for optimization of discretized nonstationary systems. It develops, for time-varying linear systems, the optimal controller gains, the optimal estimator gains, and corresponding costate and error covariance matrices. In addition, it computes the total covariance of a system with optimal estimators.

The total covariance evaluation for systems using nonoptimum estimators, involves the dynamics of cross-covariance matrix [Volume I, Equation (10.35)]. This is not included in ADAP 2.

The subroutines in the program can be classified into three groups -- basic subroutines which implement the mathematical models, subroutines which manipulate the linear data, and auxiliary subroutines for the input and output of matrix quantities and inversion of matrices, etc.

In this section, input/output information is given first; then the main program and its subroutines are described.

ADAP 2 INPUT/OUTPUT

INPUT DESCRIPTION

Input for ADAP 2 is in the form of cards and of data stored on a permanent disc file.

Card Data Input

The first group of cards to be read is cards 1-5 which provide basic program data. Their formats are shown in Table XIX.

Th[illegible]ext group of cards to be read are the nonzero elements of the matri[illegible], D_1, Q, W_1, W_2 and H_2. These cards are read by six calls to the [illegible]tine INPT. The calling statement of this subroutine is

CA[illegible] (A, NROW, NCOL)

Table XIX. Format for ADAP 2 Data Input Cards 1-5

Card/Format	Column	Quantity	Description
1/(4I2)	1-2	NX	Number of states
	3-4	NR	Number of responses
	5-6	NU	Number of controls
	7-8	NW	Number of disturbances
2/(5I4)	1-4	NRT	Number of measurements
	5-8	NM	MN = 0 Do not compute mean 1 Compute mean
	9-12	MO	Number of times through the outer loop in the subroutines GAIN and COV.
	13-16	MIL	Number of times through the inner loop in the subroutines GAIN and COV for the "last" data interval (i. e., the release time is some fraction of a regular data interval MIL/MIR.
	17-20	MIR	Number of times through the inner loop in the subroutines GAIN and COV for a "regular" data interval.
3/(4I4)	1-4	NME	NME = 0 No estimator computation 1 Estimator computation
	5-8	NRB	NRB = 1
	9-12	NRE	NRE = MR
	13-16	NFREQ	Output will occur every NFREQ times through inner loop in GAIN and COV during "regular" data intervals. During the "last" data interval output will occur at the end of MIL times through the inner loop.
4/(3I2, F10.4)	1-2	ITAPE	Logical tape number ITAPE
	3-4	NTAPE	Logical tape number NTAPE
	5-6	NDPTS	Number of data points (i. e., number of discrete times ADAP 1 outputs linear data)
	7-16	DT	Sampling time
5/(I2)	1-2	IRUN	IRUN = 0 Compute gains and performance $\neq$ 0 Read in gains and compute performance

where

A is the name of the matrix whose elements are to be read

NROW is the row dimension of the matrix A

NCOL is the column dimension of the matrix A

The format used to read the cards is (5(2I2, E12.5)). From one to five elements can be read in per card. Each element is preceded by its row and column index. The format is shown in Table XX.

Table XX. Card for Matrix Input Subroutine INPT, Format [5(2I2, E12.5)]

Column	Quantity	Description
1-2	ID(1)	Row index of 1st element
3-4	JD(1)	Column index of 1st element
5-16	YD(1)	1st element
17-18	ID(2)	Row index of 2nd element
19-20	JD(2)	Column index of 2nd element
21-32	YD(2)	2nd element
33-34	ID(3)	Row index of 3rd element
35-36	JD(3)	Column index of 3rd element
37-48	UD(3)	3rd element
49-50	ID(4)	Row index of 4th element
51-52	JD(4)	Column index of 4th element
53-64	YD(4)	4th element
65-66	ID(5)	Row index of 5th element
67-68	JD(5)	Column index of 5th element
69-80	YD(5)	5th element

A blank card is used to terminate the reading of the elements of a matrix. When the subroutine INPT senses a blank card, it returns control to the calling program. The blank card must appear in the data deck even if no data is to be read into the matrix. In other words for every call to the subroutine INPT a blank card must appear in the data deck.

Following the cards for the matrices H_1, P_1, Q, W_1, W_2, and H_2, two cards are read (see Tables XXI and XXII). These cards contain integer data which is read into the integer vector ISHUF in the subroutine DATAGEN. The vector ISHUF is used to "shuffle" the linear data after it is read from a permanent disk file designated by ITAPPF and before it is written on a

Table XXI. First Card for ISHUF, Format (21I2)

Column	Quantity	Description	
1-2	ISHUF(1)	State vector components of "standard order" which is to be	1st component
3-4	ISHUF(2)	↓	2nd component
5-6	ISHUF(3)		3rd component
7-8	ISHUF(4)		4th component
9-10	ISHUF(5)		5th component
11-12	ISHUF(6)		6th component
13-14	ISHUF(7)		7th component
15-16	ISHUF(8)		8th component
17-18	ISHUF(9)		9th component
19-20	ISHUF(10)		10th component
21-22	ISHUF(11)		11th component
23-24	ISHUF(12)		12th component
25-26	ISHUF(13)		13th component
27-28	ISHUF(14)		14th component
29-30	ISHUF(15)		15th component
31-32	ISHUF(16)		16th component
33-34	ISHUF(17)		17th component
35-36	ISHUF(18)		18th component
37-38	ISHUF(19)		19th component
39-40	ISHUF(20)		20th component
41-42	ISHUF(21)	Control vector component of "standard order" which is to be	1st component

Table XXII. Second Card for ISHUF, Format (21I2)

Column	Quantity	Description	
1-2	ISHUF(22)	Control vector component of "standard order" which is to be	2nd component
3-4	ISHUF(23)		3rd component
5-6	ISHUF(24)		4th component
7-8	ISHUF(25)		5th component
9-10	ISHUF(26)		6th component
11-12	ISHUF(27)		7th component
13-14	ISHUF(28)		8th component
15-16	ISHUF(29)	Column of mean input matrix of "standard order" which is to be	1st column
17-18	ISHUF(30)		2nd column
19-20	ISHUF(31)		3rd column
21-22	ISHUF(32)		4th column
23-24	ISHUF(33)		5th column
25-26	ISHUF(34)		6th column
27-28	ISHUF(35)	Disturbance vector component of "standard order" which is to be	1st component
29-30	ISHUF(36)		2nd component
31-32	ISHUF(37)		3rd component
33-34	ISHUF(38)		4th component
35-36	ISHUF(39)	Mean wind vector component of "standard order" which is to be	1st component
37-38	ISHUF(40)		2nd component
39-40	ISHUF(41)		3rd component

scratch disk file designated by ITAPE. Forty-one integer numbers are read into the vector ISHUF. The first twenty give the reordering of the state vector, the next eight give the reordering of the control vector, the next six give the reordering of the mean wind input matrix, the next four give the reordering of the disturbance vector, and the last three give the reordering of the mean wind input vector. All of the reordering is relative to the "standard order" which is used in the linearizing program ADAP .

The next cards in the data deck depend on the input parameter IRUN (the fifth card in the data deck). If IRUN $\neq 0$ the program expects to read in the time varying gain matrix KV backward in time. This is accomplished by reading N values of the gain matrix (where N = NDPTS, i.e., the number of data points) and writing these values on a scratch disk file designated by NTAPE. The N values represent the gain matrix KV at the following discrete time points: t_r (release time), $[t_r]$, $[t_r]-1$, $[t_r]-2$, ..., 0. (The bracket function $[X]$ is the largest integer such that $X \geq [X]$.) It should be mentioned that the program expects the gains to be in the "shuffled" order (i.e., compatible with the shuffled state and control vector).

If IRUN = 0 the program will <u>not</u> expect to read the gain matrix KV. In either case (i.e., IRUN $\neq 0$ or IRUN = 0), the last cards in the data deck will be the initial value of the state covariance matrix X_0. It should be mentioned that both the gain matrices and the state covariance matrix data is read by the subroutine INPT. Figures 48 through 55 show examples of the card data, and Figure 56 shows the entire data deck.

Permanent Disc File Data Input

The data on permanent disc file is the linear data for the a/c and is generated by program ADAP 1. The data consists of the matrices F, G_1, G_2, G_3, H_p, and V_W at the discrete times 0, 1, 2, ..., $[t_r]$, t_r (where t_r is the release time and $[X]$ is the bracket function). The total number of points is given by the input parameter NDPTS. The matrices for each data point are read from the permanent disc file designated by ITAPPF. The data is then "shuffled" and written on a scratch disc file designated by ITAPE. Access to the permanent disc file is achieved by the following control card:

```
COLUMN 1
↓
ATTACH, TAPE8, XXXX, ID = DYYYYYY.
```

where

XXXX = CATALOG NAME

YYYYYY = CATALOG NUMBER

The associated catalog control card which is used in ADAP 1 is:

CATALOG, TAPE7, XXXX, RN = 1, ID = DYYYYYY

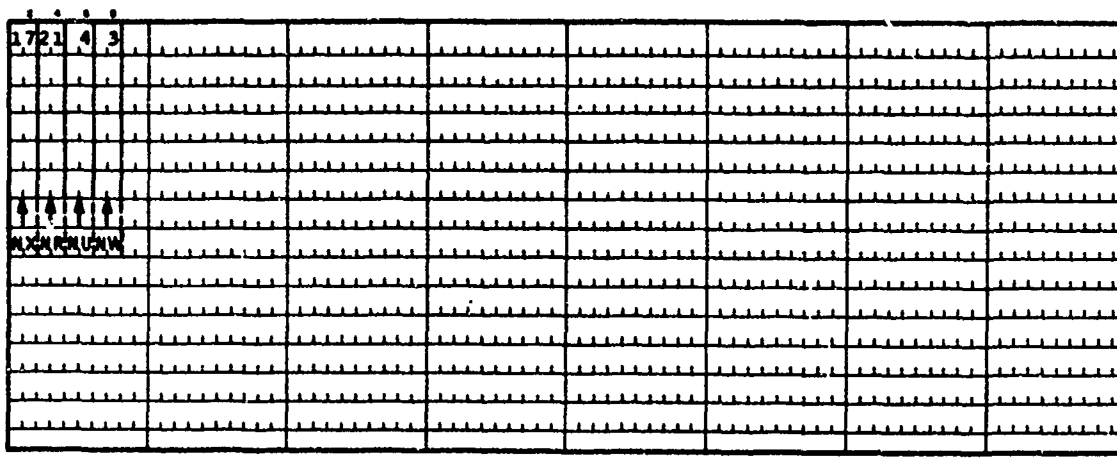


Figure 48. Data Card 1

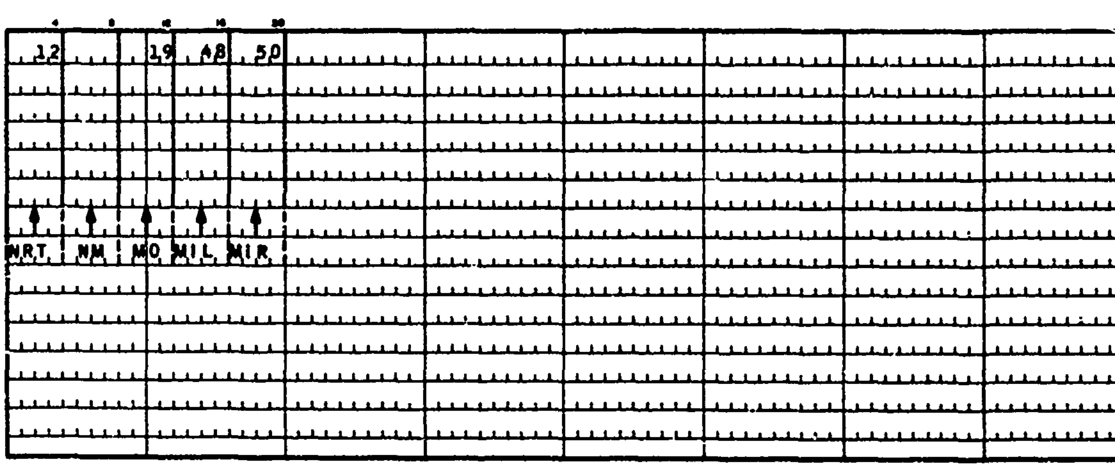


Figure 49. Data Card 2

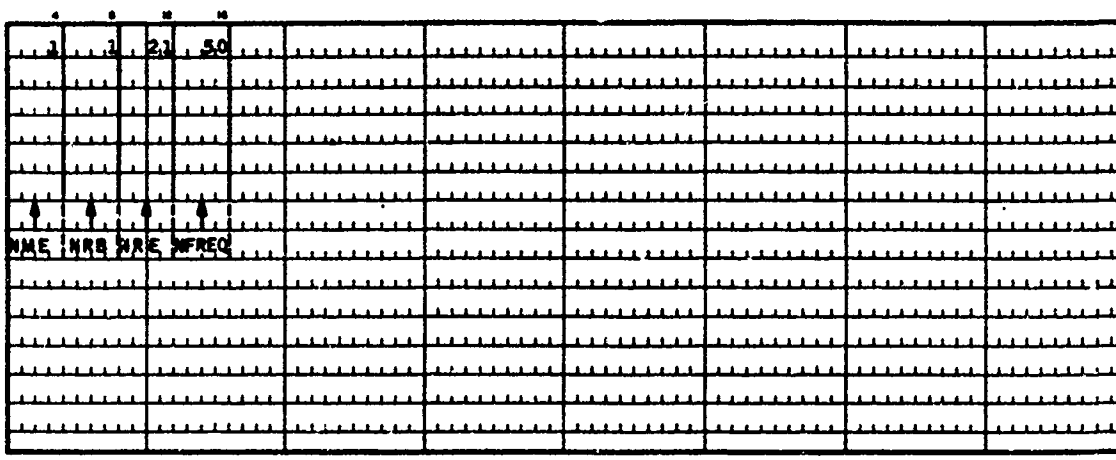


Figure 50. Data Card 3

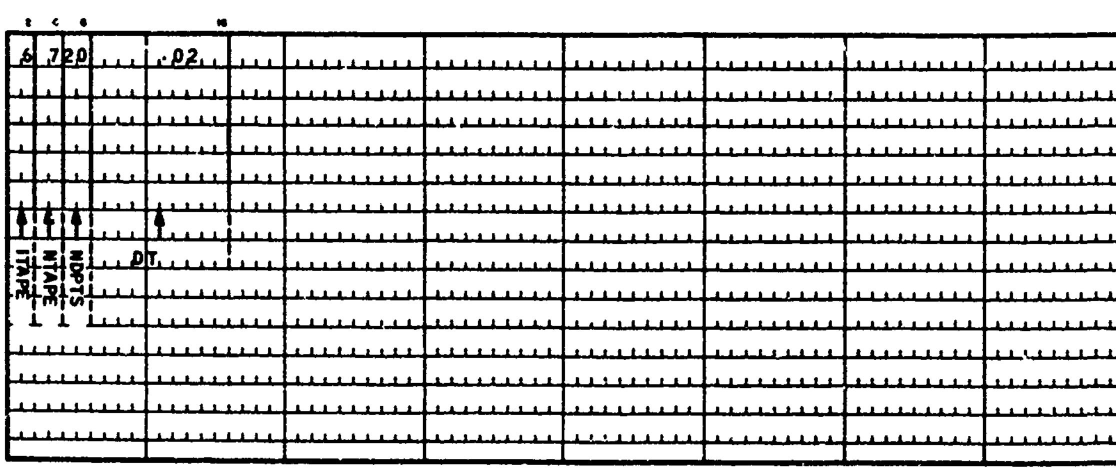


Figure 51. Data Card 4

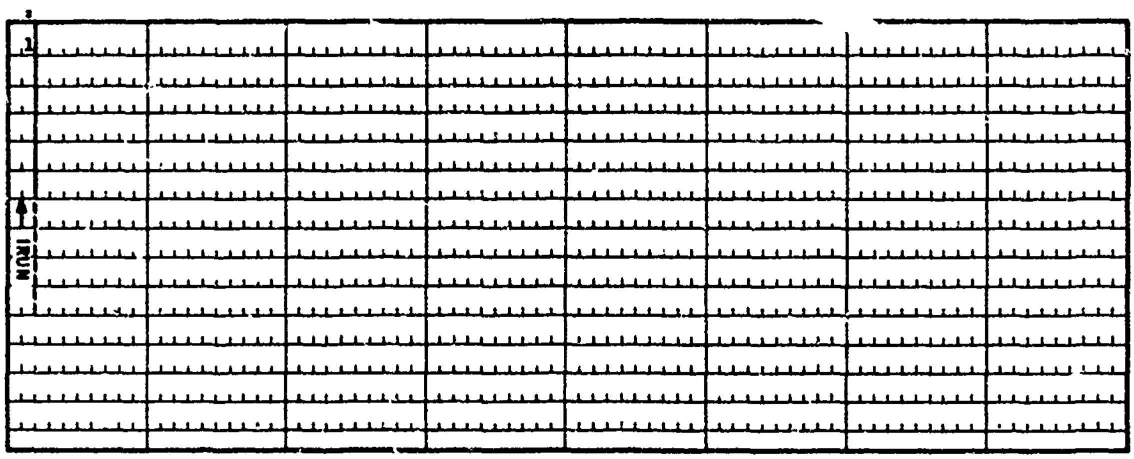


Figure 52. Data Card 5

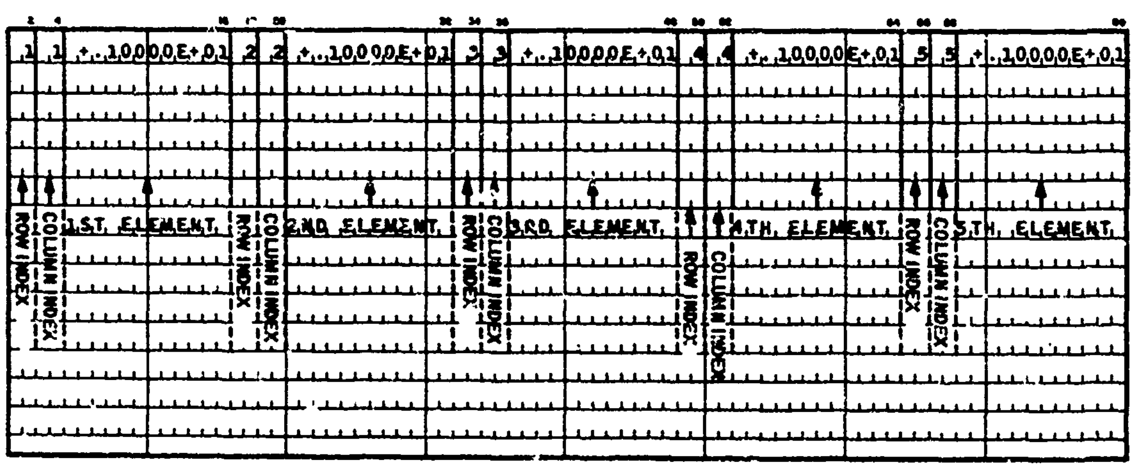


Figure 53. Example Data Card for Matrix Input Subroutine INPT

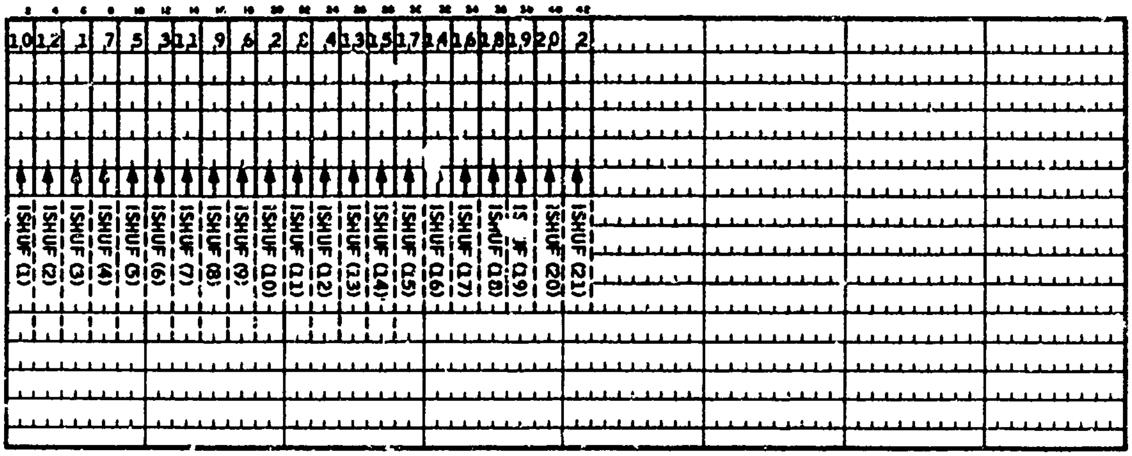


Figure 54. First Data Card for Shuffling Vector ISHUF

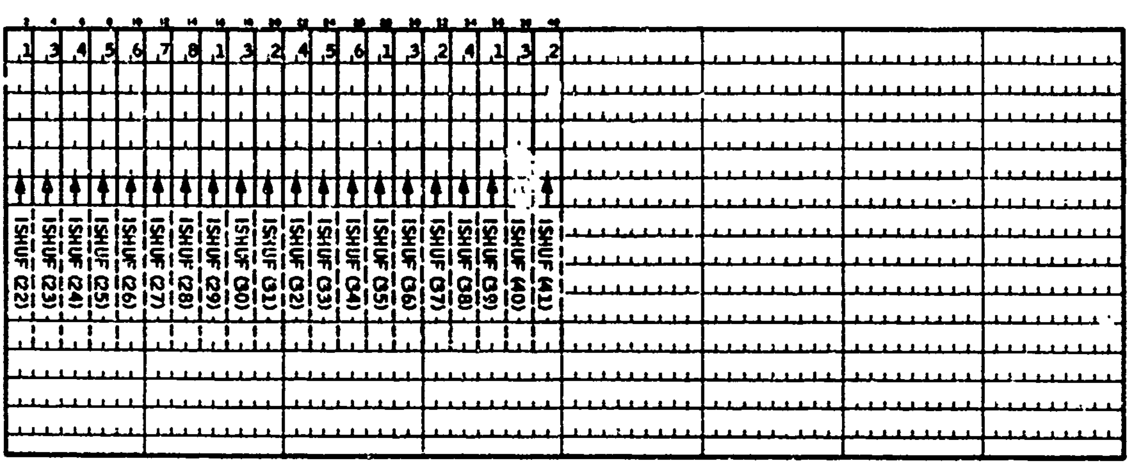


Figure 55. Second Data Card for Shuffling Vector ISHUF

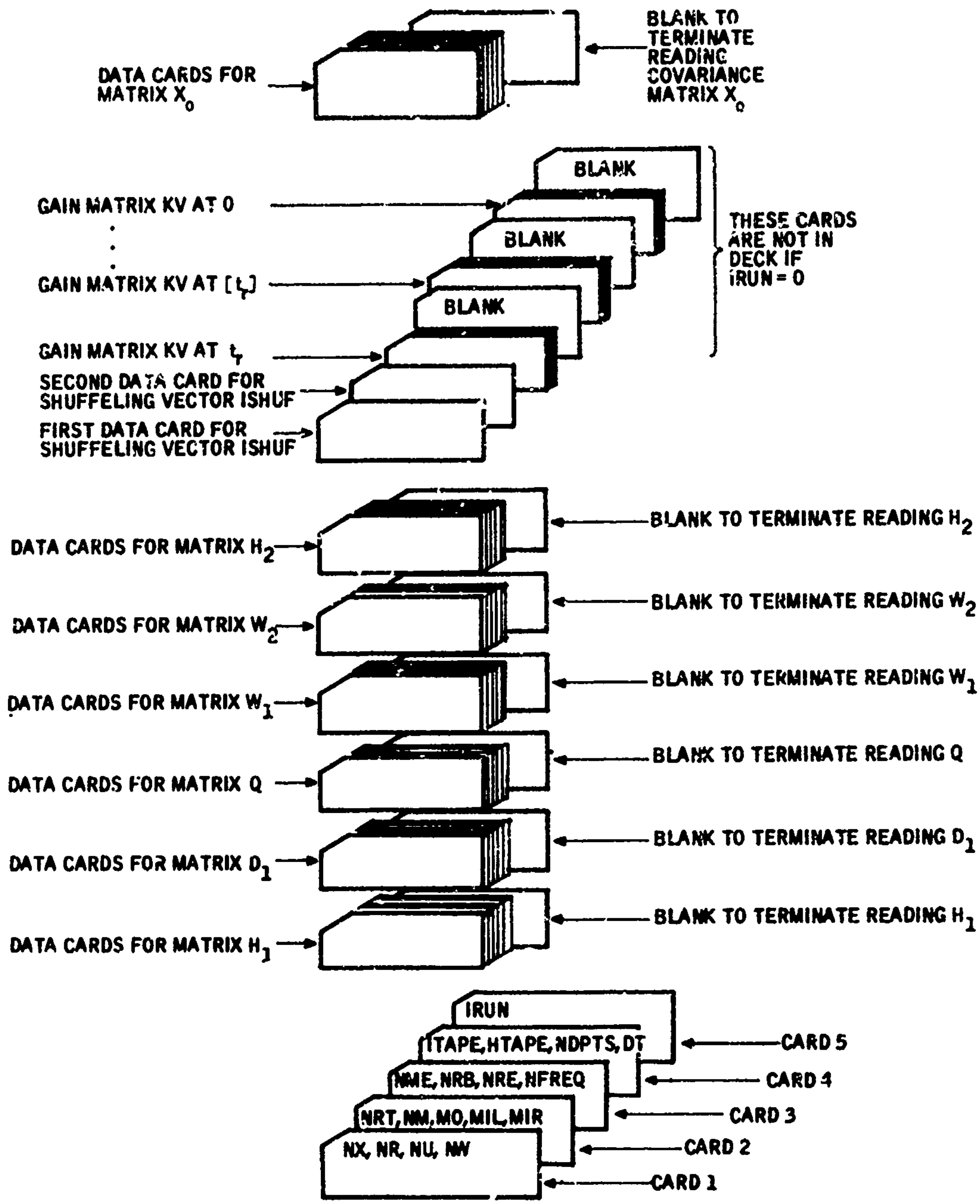


Figure 56. ADAP 2 Input Data Deck

OUTPUT DESCRIPTION

Program ADAP 2 provides both printed output and punched card output.

Printed Output

Printed output occurs in four places in program ADAP 2 -- in the main program, subroutine GAIN, subroutine COV, and subroutine ESTE.

Main Program Printed Output -- All of the input parameters on the first five input data cards are printed out. These parameters are:

Card	Parameters
1	NX, NR, NU, NW
2	NRT, NM, MO, MIL, MIR
3	NME, NRB, NRE NFREQ
4	ITAPE, NTPAE, NOPTS, DT
5	IRUN

The last printed output in the main program is the weighting matrix Q, which is printed by a call to the matrix print subroutine MP.

The calling sequence for the matrix print subroutine is

CALL MP (II, JJ, I, J, A,LW)

where

II is row dimension of the matrix A

JJ is column dimension of the matrix A

I is the number of rows of A to be printed $I \leq II$

J is the number of columns of A to be printed $J \leq JJ$

A is the name of the matrix to be printed

LW is the output tape number

GAIN Subroutine Printed Output -- If the input parameter IRUN equals zero, subroutine GAIN is entered to compute the time-varying feedback gain KV. If, in addition, the input parameter NM is not zero the deterministic input vector F will be computed. If the order of the control vector is NU and the order of the state vector is NX then the NU components of the deterministic input vector F will be stored as the NX+1 element of each row of the gain matrix KV. Printed output of the gain matrix KV and the deterministic input occurs at the following discrete times: t_r (release time), $\lceil t_r \rceil$, $\lceil t_r \rceil$-1, $[t_r]$-2, ..., 0. (i.e., the gains come out backward in time). The matrix print subroutine MP is used to print out on NU by NX+1 matrix at each of the above discrete times. The extra component in each row as explained above, is one of the NU components of the deterministic input vector.

COV Subroutine Printed Output -- If the input parameter NM is not zero, the printed output in subroutine COV consists of the diagonal elements of the response covariance matrix S, the mean response vector R, and the state covariance matrix X at the following discrete times:

1, 2, ... $[t_r]$, t_r (release time) (i.e., forward in time).

The matrix print subroutine MP is used to print the state covariance matrix X. If the input parameter NM is zero, R will not be printed.

ESTE Subroutine Printed Output -- If the input parameter NME is not zero, the subroutine ESTE is entered and the error covariance matrix PN and estimator gains L are computed and printed at the same discrete times as the diagonal elements of the response covariance matrix S and the state covariance matrix X. Both the gains L and the error covariance matrix P are printed by the matrix print subroutine MP.

Punched Card Output

Punched output occurs at two places in program ADAP 2, in subroutine GAIN and in subroutine COV.

GAIN Subroutine Punched Output -- The NU by NX+1 gain matrix K_v is punched at the same time it is printed. The subroutine OUTP is used to punch matrices. The calling sequence for the subroutine OUTP is

CALL OUTP (II, JJ, I, J, A,LP)

where

II is the row dimension of the matrix A

JJ is the column dimension of the matrix A

I is the number of rows of A to be punched

J is the number of columns of A to be punched

A is the name of the matrix to be punched

LP is the punch tape number

It should be pointed out that subroutine OUTP punches only the nonzero elements of the matrix A.

COV Subroutine Punched Output -- The value of the state covariance matrix X is punched at the release time t_r. The punch subroutine OUTP is used to punch the matrix X.

ADAP 2 PROGRAM DESCRIPTION

ADAP 2 MAIN PROGRAM

ADAP 2 is the main program for the optimization of discretized time varying systems. Its structure is very simple. At the beginning of the program some parameters are set and some parameters are read in through cards, and their contents are printed out for checking purpose. Then all matrix locations and the three matrix scratchpads are cleared. Subsequently, some of the system matrices and one-half of the weighting matrix Q are read in through cards. The other half of the symmetric weighting matrix Q is set, and the whole matrix is printed out for checking. Then the subroutine DATAGEN is called. This subroutine shuffles the linear data, and inserts some of the matrices read through card input and stores them in a scratch disc file.

At this point the linear data is ready for use. Subroutine CALLSUB is called. Depending upon whether the controller gains are to be computed or to be input through cards (IRUN flag), CALLSUB calls the pertinent subroutines. After the execution of these subroutines the program control returns to main and stops.

The flow diagram of Program ADAP 2 is shown in Figure 57 and the program listing in Figure 58. Symbols are listed in Table XXIII.

All subroutines in program ADAP 2 are listed and briefly described in Table XXIV.

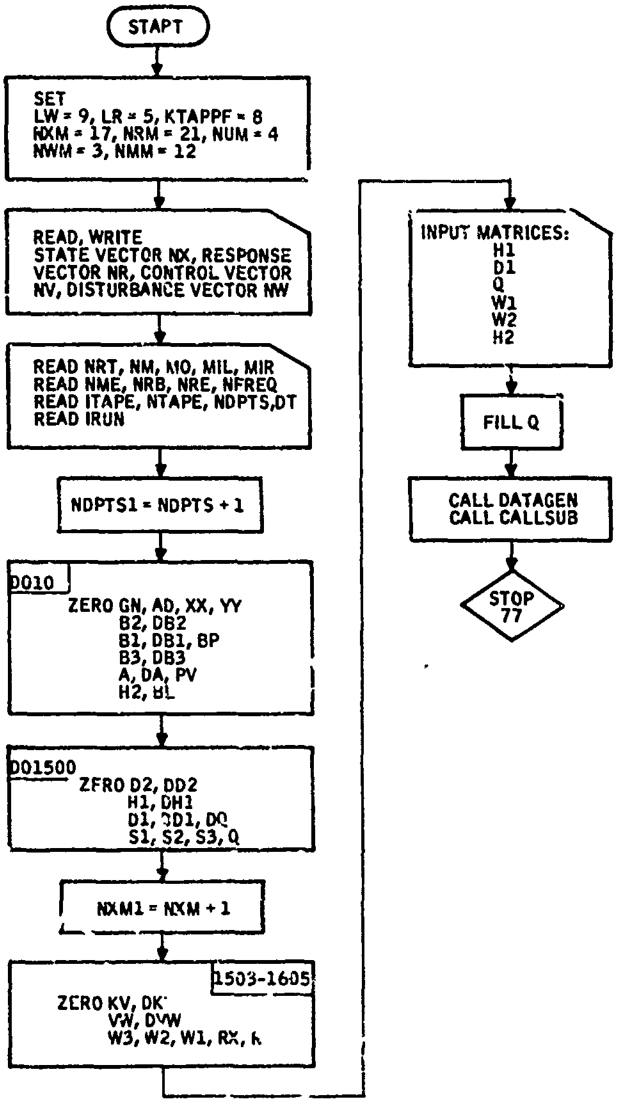


Figure 57. ADAP 2 Main Program Flow Diagram

```
      PROGRAM ADAP2(INPUT,OUTPUT,PUNCH,TAPE5=INPUT,TAPE9=OUTPUT,TAPE3=PU
     1NCH,TAPE4,TAPE6,TAPE7,TAPE8)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      LW=9
      LR=5
      KTAPE=4
      ITAPPF=8
      NXM=17
      NRM=21
      NUM=4
      NWM=3
      NMM=12
      READ(LR,1)NX,NR,NU,NW
    1 FORMAT(4I2)
      READ(LR,3)NRT,NM,MO,MIL,MIR
    3 FORMAT(5I4)
      READ(LR,4)NME,NRB,NRE,NFREQ
    4 FORMAT(4I4)
      READ(LR,5)ITAPE,NTAPE,NDPTS,DT
      READ(LR,20)IRUN
   20 FORMAT(I2)
      WRITE(LW,1930)NX,NR,NU,NW
      WRITE(LW,1931)NRT,NM,MO,MIL,MIR
      WRITE(LW,1932)NME,NRB,NRE,NFREQ
      WRITE(LW,1933)ITAPE,NTAPE,NDPTS,DT
      WRITE(LW,1934)IRUN
 1930 FORMAT(1H1/7X,4H NX=I2,4H NR=I2,4H NU=I2,4H NW=I2//)
 1931 FORMAT(7X,5H NRT=I2,4H NM=I2,4H MO=I2,5H MIL=I2,5H MIR=I2//)
 1932 FORMAT(7X,5H NME=I2,5H NRB=I2,5H NRE=I2,7H NFREQ=I2//)
 1933 FORMAT(7X,7H ITAPE=I2,7H NTAPE=I2,7H NDPTS=I2,4H DT=F10.4//)
 1934 FORMAT(7X,6H IRUN=I2)
      NDPTS1=NDPTS+1
    5 FORMAT(3I2,F10.4)
      DO 10 I=1,NXM
      GN(I)=0.
      AD(I)=0.
      XX(I)=0.
      YY(I)=0.
      DO 1700 J=1,3
      B2(I,J)=0.
      DB2(I,J)=0.
 1700 CONTINUE
      DO 11 J=1,NUM
      B1(I,J)=0.
      DB1(I,J)=0.
```

Figure 58. ADAP 2 Main Program Input/Output Listing

```
      BP(J,I)=0.
   11 CONTINUE
      DO 1600 J=1,NWM
      B3(I,J)=0.
      DB3(I,J)=0.
 1600 CONTINUE
      DO 12 J=1,NXM
      A(I,J)=0.
      DA(I,J)=0.
      PV(I,J)=0.
   12 CONTINUE
      DO 1601 J=1,NRT
      H2(J,I)=0.
      BL(I,J)=0.
 1601 CONTINUE
   10 CONTINUE
      DO 1500 I=1,NRM
      DO 1701 J=1,3
      D2(I,J)=0.
      DD2(I,J)=0.
 1701 CONTINUE
      DO 1717 J=1,NXM
      H1(I,J)=0.
      DH1(I,J)=0.
 1717 CONTINUE
      DO 1501 J=1,NUM
      D1(I,J)=0.
      DD1(I,J)=0.
      DQ(J,I)=0.
 1501 CONTINUE
      DO 1503 J=1,NRM
      S1(I,J)=0.
      S2(I,J)=0.
      S3(I,J)=0.
      Q(I,J)=0.
 1503 CONTINUE
 1500 CONTINUE
      NXM1=NXM+1
      DO 1606 J=1,NUM
      DO 1606 I=1,NXM1
      KV(J,I)=0.
      DKV(J,I)=0.
 1606 CONTINUE
      DO 1607 I=1,3
      VW(I)=0.
      DVW(I)=0.
 1607 CONTINUE
      DO 1604 I=1,NWM
      DO 1604 J=1,NRT
      W3(I,J)=0.
 1604 CONTINUE
      DO 1603 I=1,NRT
      DO 1603 J=1,NRT
      W2(I,J)=0.
 1603 CONTINUE
      DO 22 I=1,NWM
      DO 22 J=1,NWM
   22 W1(I,J)=0.
      DO 1605 I=1,NRM
      RX(I,1)=0.
      R(I,1)=0.
 1605 CONTINUE
      CALL INPT(H1,NRM,NXM)
      CALL INPT(D1,NRM,NUM)
      CALL INPT(Q,NRM,NRM)
      CALL INPT(W1,NWM,NWM)
      CALL INPT(W2,NMM,NMM)
      CALL INPT(H2,NMM,NXM)
      DO 1702 I=1,NRE
      DO 1702 J=I,NRE
      Q(J,I)=Q(I,J)
 1702 CONTINUE
      WRITE(LW,1801)
 1801 FORMAT(1H1/7X,17H WEIGHTING MATRIX/)
      CALL MP(NRM,NRM,NR,NR,Q,LW)
      CALL DATAGEN(ITAPPF,NDPTS)
      CALL CALLSUB(NDPTS,KTAPE,IRUN)
      STOP 77
      END
```

Figure 58. ADAP 2 Main Program Input/Output Listing (concluded)

Table XXIII. List of Symbols for ADAP 2 (DISCOP)

Mnemonic	Value	Input	Description
LW	9		Output tape number - set in Main
LR	5		Input tape number - set in Main
KTAPE	4		Scratch tape number - set in Main
ITAPPF	8		Permanent file tape number - set in Main
NXM	17		Maximum state size - set in Main
NRM	21		Maximum response size - set in Main
NUM	4		Maximum control size - set in Main
NWM	3		Maximum disturbance size - set in Main
NMM	12		Maximum measurement size - set in Main
NX	17	X	State vector size
NR	21	X	Response vector size
NU	4	X	Control vector size
NW	3	X	Disturbance vector size
NRT	12	X	Measurement vector size
NM	0	X	NM = 0 ⇒ no mean computation; NM ≠ 0 ⇒ mean computation
$\bar{\text{MO}}$	19	X	Number of times outer loop is executed in subroutines COV and GAIN
MIL	48	X	Number of times inner loop is executed for last data point in subroutines COV and GAIN

Table XXIII. List of Symbols for ADAP 2 (DISCOP) (continued)

Mnemonic	Value	Input	Description
MIR	50	X	Number of times inner loop is executed for all data points except the last in subroutines COV and GAIN
NME	1	X	NME = 0 ⇒ no optimal estimator; NME ≠ 0 ⇒ optimal estimator
NRB	1	X	Used to calculate terminal value of $P_v(N)$ and G(N)
NRE	17	X	
NRFEQ	50	X	Frequency of output in inner loop, i.e., output every NFREQ inner loop
ITAPE	6	X	Logical tape number for time-varying coefficient data
NTAPE	7	X	Logical tape number for time-varying gains data
NDPTS	20	X	Total number of data points
DT	0.02	X	Step size
IRUN		X	IRUN = 0 ⇒ compute control gains and compute performance; IRUN ≠ 0 ⇒ read control gains and computer performance
A			17 x 17 array which contains current value of $[I+\Delta t\ F(t)]$
DA			17 x 17 array which contains current value of difference for $\Delta t\ F(t)$
B1			17 x 4 array which contains current value of $\Delta t\ B_1(t)$ (i.e., control input matrix)
DB1			17 x 4 array which contains value of difference for $\Delta t\ B_1(t)$

Table XXIII. List of Symbols for ADAP 2 (DISCOP) (continued)

Mnemonic	Value	Input	Description
B2			17 x 3 array which contains current value of $\Delta t\ B_2(t)$ (i. e., mean input matrix)
DB2			17 x 3 array which contains current value of difference for $\Delta t\ B_2(t)$
B3			17 x 3 array which contains current value of $\Delta t\ B_3(t)$ (i. e., disturbance input matrix)
DB3			17 x 3 array which contains current value of difference for $\Delta t\ B_3(t)$
H1		X	21 x 17 array which contains current value of $H_1(t)$ (i. e., matrix multiplied by state x in response equation
DH1			21 x 17 array which contains current value of difference for $H_1(t)$
D1		X	21 x 4 array which contains current value of $D_1(t)$ (i. e., control input matrix in response equation)
DD1			21 x 4 array which contains current value of difference for $D_1(t)$
D2		X	21 x 3 array which contains current value of $D_2(t)$ (i. e., mean input matrix for response equation)
DD2			21 x 3 array which contains current value of difference for $D_2(t)$
PV			17 x 17 array which contains current value of P_v in subroutine GAIN and P_η in subroutine COV

Table XXIII. List of Symbols for ADAP 2 (DISCOP) (continued)

Mnemonic	Value	Input	Description
KV			4 x 18 array which contains current value of $K_v(t)$ (i. e., gain matrix)
FV			4-vector which contains current value of deterministic input
GN			17-vector which contains current value of G(N)
AD			17-vector which contains current value of diagonal elements of $\Delta t\ F(t)$
VW			3-vector which contains current value of mean wind input
S1			21 x 21 working array
S2			21 x 21 working array
S3			21 x 21 working array
BP			4 x 17 array which contains the matrix product B_1P_v
DQ			4 x 21 array which contains the matrix product $D_1'Q$
BPB			4 x 4 array which contains the matrix product $B_1'P_v B_1$
DQD			4 x 4 array which contains the matrix product $D_1'QD_1$
SUM			4 x 4 array which contains the matrix sum $(B_1' P_vB_1 + \Delta t\ D_1'QD)$
KWA			4-vector used to invert $(B_1'P_vB_1 + \Delta tD_1'QD_1)$

Table XXIII. List of Symbols for ADAP 2 (DISCOP) (concluded)

Mnemonic	Value	Input	Description
$\dot{Q}$			21 x 21 array which contains the quadratic weights
DKV			4 x 18 array which contains current value of difference for gains K_V
XX			17-vector which contains current value of mean state
YY			17-vector which is used to compute mean state
X			17 x 17 array which contains current value of state covariance matrix
RX			21-vector which contains current value of response covariance (diagonal elements)
R			21-vector which contains current value of mean response
W1			3 x 3 array which is the expected value of the disturbance input
W2		X	12 x 12 array which is the expected value of the measurement error
W3			3 x 12 array
H2		X	12 x 17 array measurement matrix
DFV			4-vector which contains current value of the difference for the deterministic input
DVW			3-vector which contains the current value of the difference for the mean wind VW
BL			17 x 12 array which contains the current value of the estimator gains L

Table XXIV. ADAP 2 (DISCOP) Subroutine Summary

Subroutine	Description	Flow Diagram (Figure No.)	Program Listing (Figure No.)	List of Symbols (Table No.)	Volume I Reference (pp.)
ADAP 2 (DISCOP) (Main Program)	Main program for optimization of discretized nonstationary systems	57	58	XXIII	149-163
CALLSUB	Subroutine caller	59	60	---	---
GAIN	Nonstationary costate and deterministic input dynamics and nonstationary controller gains	61	62	---	158-159
COV	Nonstationary state covariance and mean-response dynamics	63	64	---	160-162
ESTE	Nonstationary estimation error covariance dynamics and nonstationary estimator gains	65	66	---	159-160
DATAGEN	Shuffled linear data generator	67	68	---	104-110
SHUF	Shuffler	69	70	---	104-110
REVS	Linear data reverser	71	72	---	---
DIFBC	Current differences of linear data	73	74	---	---
BCOEF	Current value of backward-time linear data	75	76	---	---
DIFG	Controller gain difference	77	78	---	---
RDWT	Read-write tape data transfer	79	80	---	---
REVG	Controller gain reverser	81	82	---	---
FCOEF	Current value of forward-time linear data	83	84	---	---
MP	Matrix print	85	86	---	---
INPT	Matrix input	8[illegible]	88	---	---
OUTP	Matrix card punch	89	90	---	---
TDINVR	Matrix invert	---	91	---	---

ADAP 2 BASIC SUBROUTINES

Subroutine CALLSUB

Subroutine CALLSUB calls all the pertinent subroutines depending upon the flag IRUN. If IRUN = 0 it computes optimal controller gains. If IRUN ≠ 0, it bypasses the controller gain computations and the necessary data manipulations for it. It reads the gain matrix through cards. The subroutine flow chart is shown in Figure 59 and its program listing is Figure 60.

Subroutine GAIN

Subroutine GAIN implements the analysis given in Section IX of Volume 1. It generates the costate matrix, the mean control input and the optimal controller gains as a function of backward time. There are two basic loops in it: integration loop (DO 200), and data update loop (DO 100).

At each data point including the release time data point the gains are printed out and also are punched into cards.

The subroutines flow chart is shown in Figure 61 and the program listing in Figure 62.

Subroutine COV

Subroutine COV implements the analysis given in Section IX of Volume I. It generates the estimation error covariance matrix, the optimal estimator gains, the mean response and the state covariance and the response covariance as a function of forward time. It prints out these quantities at three specified time points. At the final time point (i. e., the release time) the the state covariance matrix is punched into cards for subsequent use by ADAP3. The estimator gains and the estimation error covariance are obtained by calling subroutine ESTE if the flag NME ≠ 0. It has two major loops, data update loop (DO 100) and integration loop (DO 200).

The subroutine COV flow chart is shown in Figure 63 and the program listing in Figure 64.

Subroutine ESTE

Subroutine ESTE implements the analysis given in Section IX of Volume I. It generates the covariance of estimation error and the optimal estimator gains it prints out the estimator gains. Its flow chart is given in Figure 65 and its program listing in Figure 66.

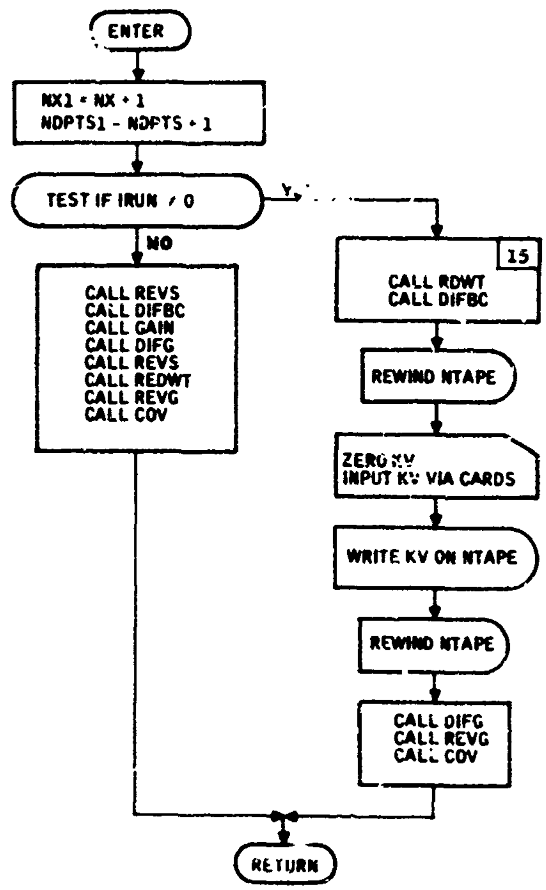


Figure 59. Subroutine CALLSUB Flow Diagram

```
      SUBROUTINE CALLSUB(NDPTS,KTAPE,IRUN)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      NX1=NX+1
      NDPTS1=NDPTS+1
      IF(IRUN.NE.0) GOTO 15
      CALL REVS(NDPTS)
      CALL DIFBC(NDPTS,0)
      CALL GAIN
      CALL DIFG(KTAPE,NDPTS)
      CALL REVS(NDPTS1)
      CALL RDWT(NTAPE,ITAPE,NDPTS1)
      CALL REVG(KTAPE,NTAPE,NDPTS1)
      CALL COV
      RETURN
   15 CONTINUE
      CALL RDWT(ITAPE,NTAPE,NDPTS)
      CALL DIFBC(NDPTS,1)
      REWIND NTAPE
      DO 11 L=1,NDPTS
      DO 10 I=1,NU
      DO 10 J=1,NX1
   10 KV(I,J)=0.
      CALL INPT(KV,4,18)
      WRITE(NTAPE)((KV(I,J),J=1,NX1),I=1,NU)
   11 CONTINUE
      REWIND NTAPE
      CALL DIFG(KTAPE,NDPTS)
      CALL REVG(KTAPE,NTAPE,NDPTS1)
      CALL COV
      RETURN
      END
```

Figure 60. Subroutine CALLSUB Program Listing

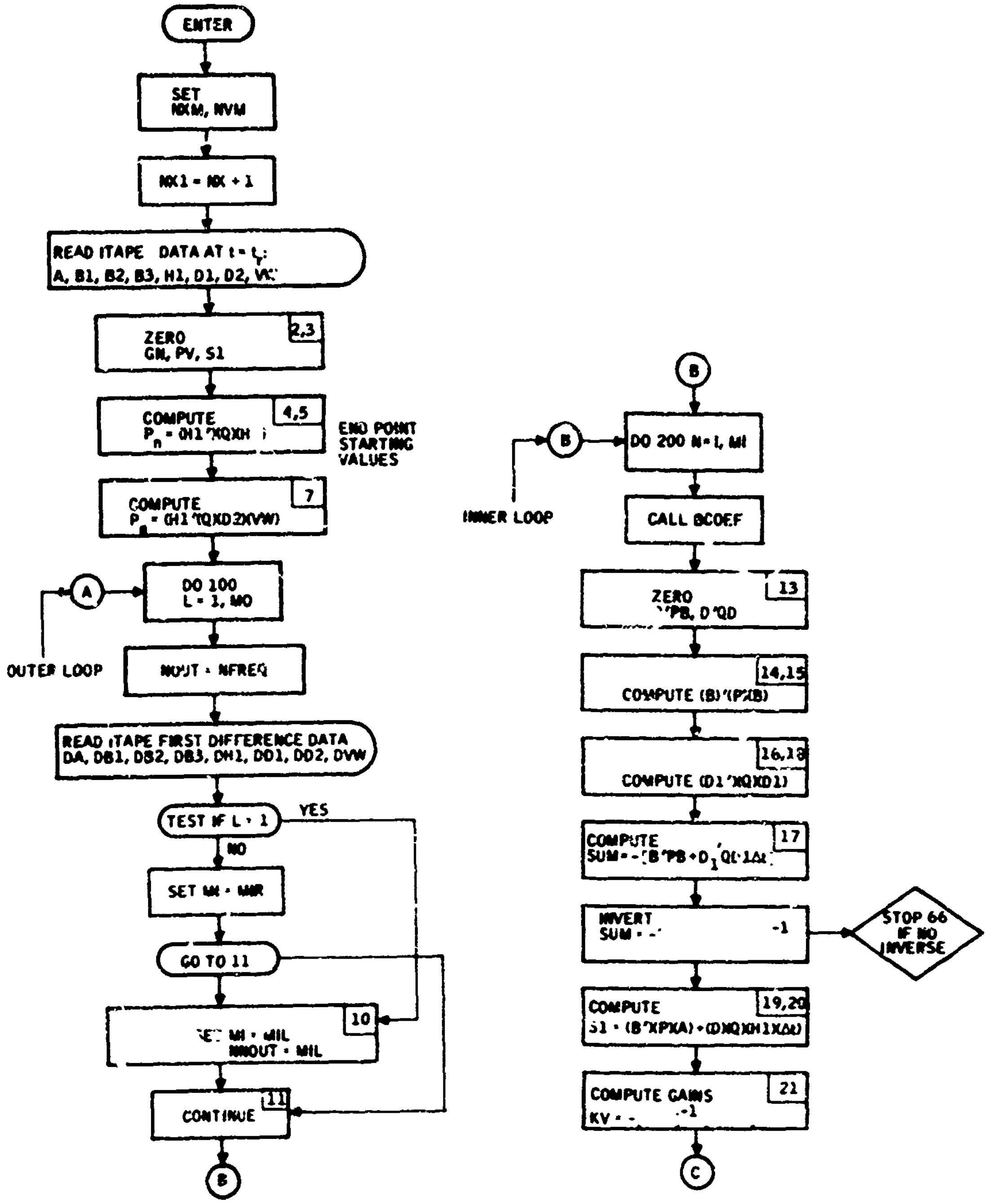


Figure 61. Subroutine GAIN Flow Diagram

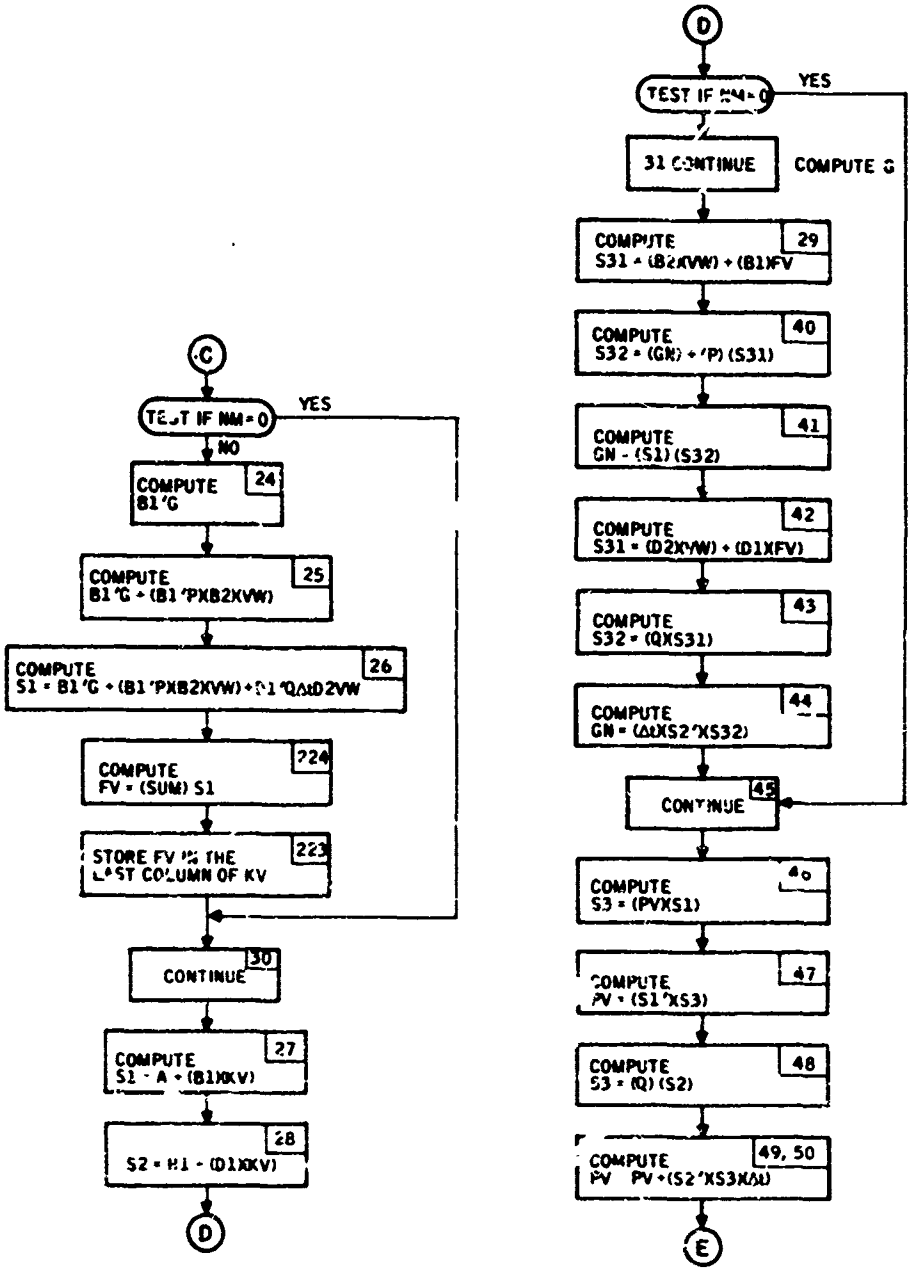


Figure 61. Subroutine GAIN Flow Diagram (continued)

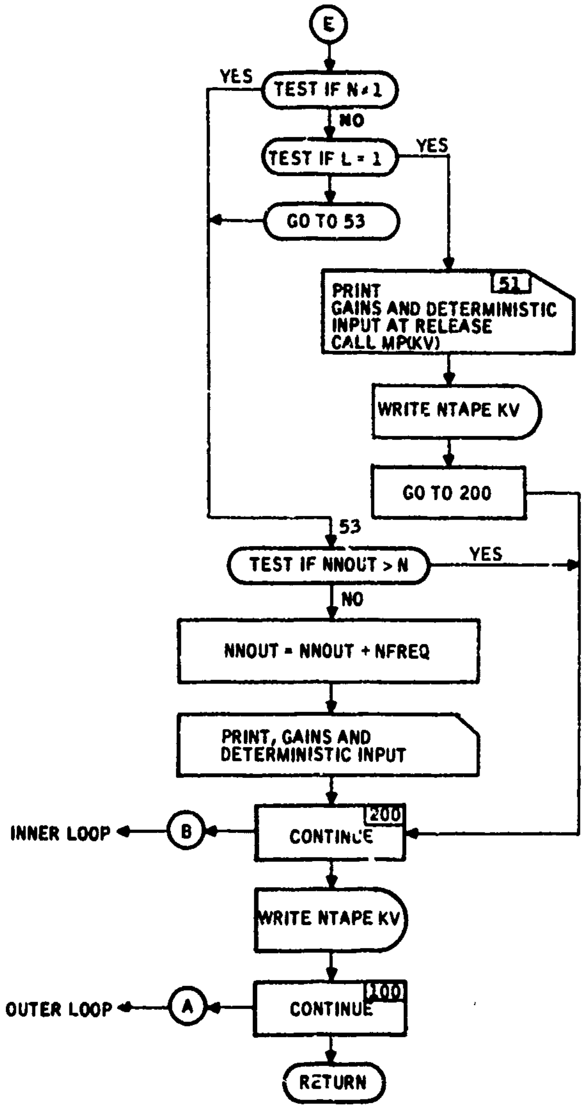


Figure 61. Subroutine GAIN Flow Diagram (concluded)

```
      SUBROUTINE GAIN
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      LP=3
      NXM=18
      NX1=NX+1
      NUM=4
C
C READ IN TERMINAL DATA AND FIRST DIFFERENCE
C
      READ(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      READ(ITAPE)((B3(I,J),J=1,NW),I=1,NX)
      READ(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      READ(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      READ(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      READ(ITAPE)VW
C
C  COMPUTE TERMINAL P AND G
      DO 2 I=1,NX
      GN(I)=0.
      DO 2 J=1,NX
    2 PV(I,J)=0.
      DO 3 I=1,NR
      DO 3 J=1,NR
    3 S1(I,J)=0.
      DO 4 I=1,NX
      DO 4 J=NRB,NRE
      DO 4 K=NRB,NRE
    4 S1(I,J)=S1(I,J)+H1(K,I)*Q(K,J)
      DO 5 I=1,NX
      DO 5 J=1,NX
      DO 5 K=NRB,NRE
    5 PV(I,J)=PV(I,J)+S1(I,K)*H1(K,J)
      DO 7  I=1,NX
      DO 7  K=NRB,NRE
    7 GN(I)=GN(I)+S1(I,K)*(D2(K,1)*VW(1)+D2(K,2)*VW(2)+D2(K,3)*VW(3))
      DO 100 L=1,MO
      NNOUT=NFREQ
      READ(ITAPE)((DA(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((DB1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((DB2(I,J),J=1,3),I=1,NX)
      READ(ITAPE)((DB3(I,J),J=1,NW),I=1,NX)
      READ(ITAPE)((DH1(I,J),J=1,NX),I=1,NRE)
      READ(ITAPE)((DD1(I,J),J=1,NU),I=1,NRE)
```

Figure 62. Subroutine GAIN Program Listing

```
      READ(ITAPE)((DD2(I,J),J=1,3),I=1,NRE)
      READ(ITAPE)DVW
      IF( L.EQ.1) GOTO 10
      MI=MIP
      GOTO 11
   10 MI=MIL
      NNOUT=MIL
   11 CONTINUE
      DO 200 N=1,MI
   12 CALL RCOEF(L,N)
      DO 13 I=1,NU
      DO 13 J=1,NU
      BPB(I,J)=0.
   13 DQD(I,J)=0.
      DO 14 I=1,NU
      DO 14 J=1,NX
      BP(I,J)=0.
      DO 14 K=1,NX
   14 BP(I,J)=BP(I,J)+B1(K,I)*PV(K,J)
      DO 15 I=1,NU
      DO 15 J=1,NU
      DO 15 K=1,NX
   15 BPB(I,J)=BPB(I,J)+BP(I,K)*B1(K,J)
      DO 16 I=1,NU
      DO 16 J=1,NR
      DQ(I,J)=0.
      DO 16 K=1,NR
   16 DQ(I,J)=DQ(I,J)+D1(K,I)*Q(K,J)
      DO 17 I=1,NU
      DO 17 J=1,NU
      DO 18 K=1,NR
   18 DQD(I,J)=DQD(I,J)+DQ(I,K)*D1(K,J)
   17 SUM(I,J)=-BPB(I,J)-DQD(I,J)*DT
C
C INVERT (BPB+DQD(DT)
C
      CALL TDINVR(ISOL,IDSOL,NU,NU,SUM,NUM,KWA,DET)
      IF((ISOL+IDSOL).GT.2) GOTO 501
      GOTO 502
  501 STOP 66
  502 CONTINUE
C
C  COM JTE GAINS KV
C
      DO 19 I=1,NU
      DO 19 J=1,NX
      S1(I,J)=0.
      DO 20 K=1,NX
   20 S1(I,J)=S1(I,J)+BP(I,K)*A(K,J)
      DO 19 K=1,NR
   19 S1(I,J)=S1(I,J)+DT*DQ(I,K)*H1(K,J)
      DO 21 I=1,NU
      DO 21 J=1,NX
      KV(I,J)=0.
```

Figure 62. Subroutine GAIN Program Listing (continued)

```
      DO 21 K=1,NU
   21 KV(I,J)=KV(I,J)+SUM(I,K)*S1(K,J)
C
C COMPUTE DETERMINISTIC INPUT FV
C
      IF(NM)22,30,22
   22 DO 23 I=1,NU
      S1(I,1)=0.
      DO 24 J=1,NX
   24 S1(I,1)=S1(I,1)+B1(J,I)*GN(J)
      DO 25 J=1,NX
   25 S1(I,1)=S1(I,1)+BP(I,J)*(B2(J,1)*VW(1)+B2(J,2)*VW(2)+B2(J,3)*VW(3)
     1)
      DO 26 J=1,NR
   26 S1(I,1)=S1(I,1)+DT*DQ(I,J)*(D2(J,1)*VW(1)+D2(J,2)*VW(2)+D2(J,3)*VW
     1(3))
   23 CONTINUE
      DO 223 I=1,NU
      FV(I)=0.
      DO 224 J=1,NU
  224 FV(I)=FV(I)+SUM(I,J)*S1(J,1)
  223 KV(I,NX1)=FV(I)
C FORM (A+BK) AND (H+DK)
   30 CONTINUE
      DO 27 I=1,NX
      DO 27 J=1,NX
      S1(I,J)=A(I,J)
      DO 27 K=1,NU
   27 S1(I,J)=S1(I,J)+B1(I,K)*KV(K,J)
      DO 28 I=1,NR
      DO 28 J=1,NX
      S2(I,J)=H1(I,J)
      DO 28 K=1,NU
   28 S2(I,J)=S2(I,J)+D1(I,K)*KV(K,J)
      IF(NM)31,45,31
C COMPUTE G(N)
   31 CONTINUE
      DO 29 I=1,NX
      S3(I,1)=B2(I,1)*VW(1)+B2(I,2)*VW(2)+B2(I,3)*VW(3)
      DO 29 J=1,NU
   29 S3(I,1)=S3(I,1)+B1(I,J)*FV(J)
      DO 40 I=1,NX
      S3(I,2)=GN(I)
      DO 40 J=1,NX
   40 S3(I,2)=S3(I,2)+PV(I,J)*S3(J,1)
      DO 41 I=1,NX
      GN(I)=0.
      DO 41 J=1,NX
   41 GN(I)=GN(I)+S1(J,I)*S3(J,2)
      DO 42 I=1,NR
      S3(I,1)=D2(I,1)*VW(1)+D2(I,2)*VW(2)+D2(I,3)*VW(3)
      DO 42 J=1,NU
   42 S3(I,1)=S3(I,1)+D1(I,J)*FV(J)
      DO 43 I=1,NR
```

Figure 62. Subroutine GAIN Program Listing (continued)

```
      S3(I,2)=0.
      DO 43 J=1,NR
   43 S3(I,2)=S3(I,2)+Q(I,J)*S3(J,1)
      DO 44 I=1,NX
      DO 44 J=1,NR
   44 GN(I)=GN(I)+DT*S2(J,I)*S3(J,2)
   45 CONTINUE
C
C COMPUTE COSTATE MATRIX PV(N)
C
      DO 46 I=1,NX
      DO 46 J=1,NX
      S3(I,J)=0.
      DO 46 K=1,NX
   46 S3(I,J)=S3(I,J)+S1(K,I)*PV(K,J)
      DO 47 I=1,NX
      DO 47 J=I,NX
      PV(I,J)=0.
      DO 47 K=1,NX
   47 PV(I,J)=PV(I,J)+S3(I,K)*S1(K,J)
      DO 48 I=1,NR
      DO 48 J=1,NX
      S3(I,J)=0.
      DO 48 K=1,NR
   48 S3(I,J)=S3(I,J)+Q(I,K)*S2(K,J)
      DO 49 I=1,NX
      DO 49 J=I,NX
      DO 50 K=1,NR
   50 PV(I,J)=PV(I,J)+S2(K,I)*S3(K,J)*DT
   49 PV(J,I)=PV(I,J)
      IF(N.NE.1) GOTO 53
      IF(L.EQ.1) GOTO 51
      GOTO 53
   51 WRITE(LW,52)
   52 FORMAT(1H1/7X,41H GAINS AND DETERMINISTIC INPUT AT RELEASE/)
      CALL MP(NUM,NXM,NU,NX1,KV,LW)
      WRITE(NTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      CALL OUTP(4,18,NU,NX1,KV,LP)
      GOTO 200
   53 IF(NNOUT.GT.N) GOTO 200
      NNOUT=NNOUT+NFREQ
      WRITE(LW,54)
   54 FORMAT(1H1/7X,30H GAINS AND DETERMINISTIC INPUT/)
      CALL MP(NUM,NXM,NU,NX1,KV,LW)
  200 CONTINUE
      WRITE(NTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      CALL OUTP(4,18,NU,NX1,KV,LP)
  100 CONTINUE
      RETURN
      END
```

Figure 62. Subroutine GAIN Program Listing (concluded)

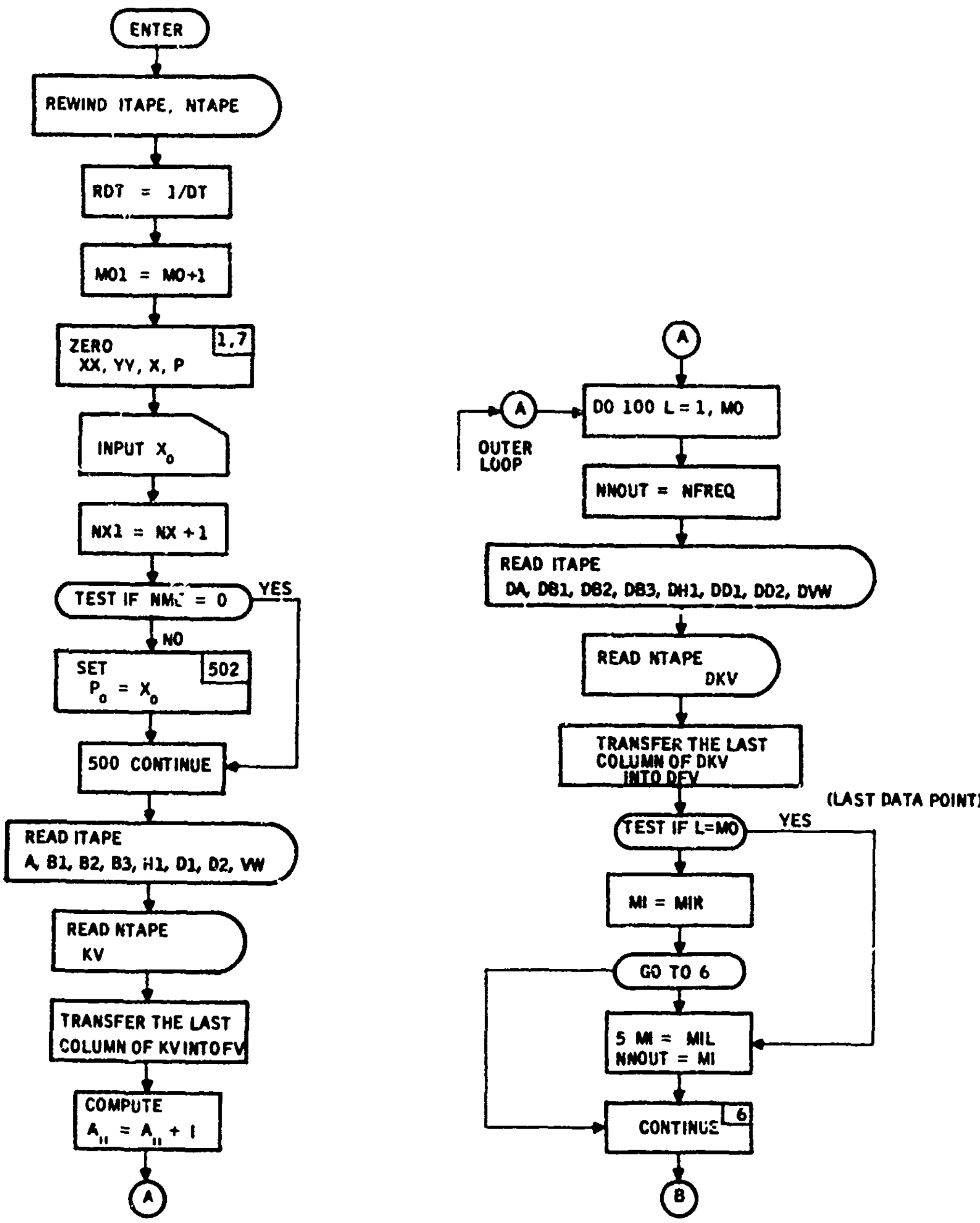


Figure 63. Subroutine COV Flow Diagram

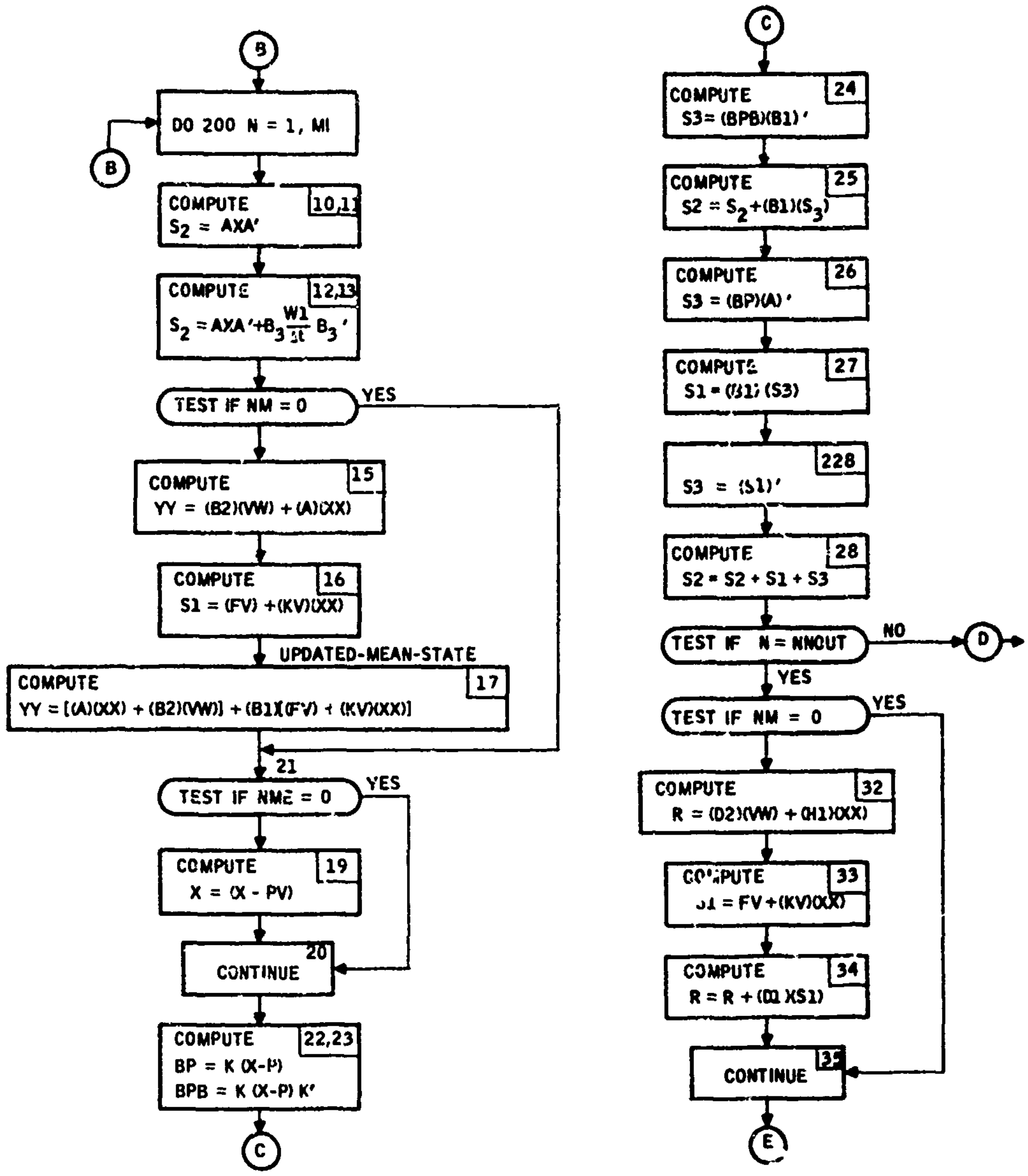


Figure 63. Subroutine COV Flow Diagram (continued)

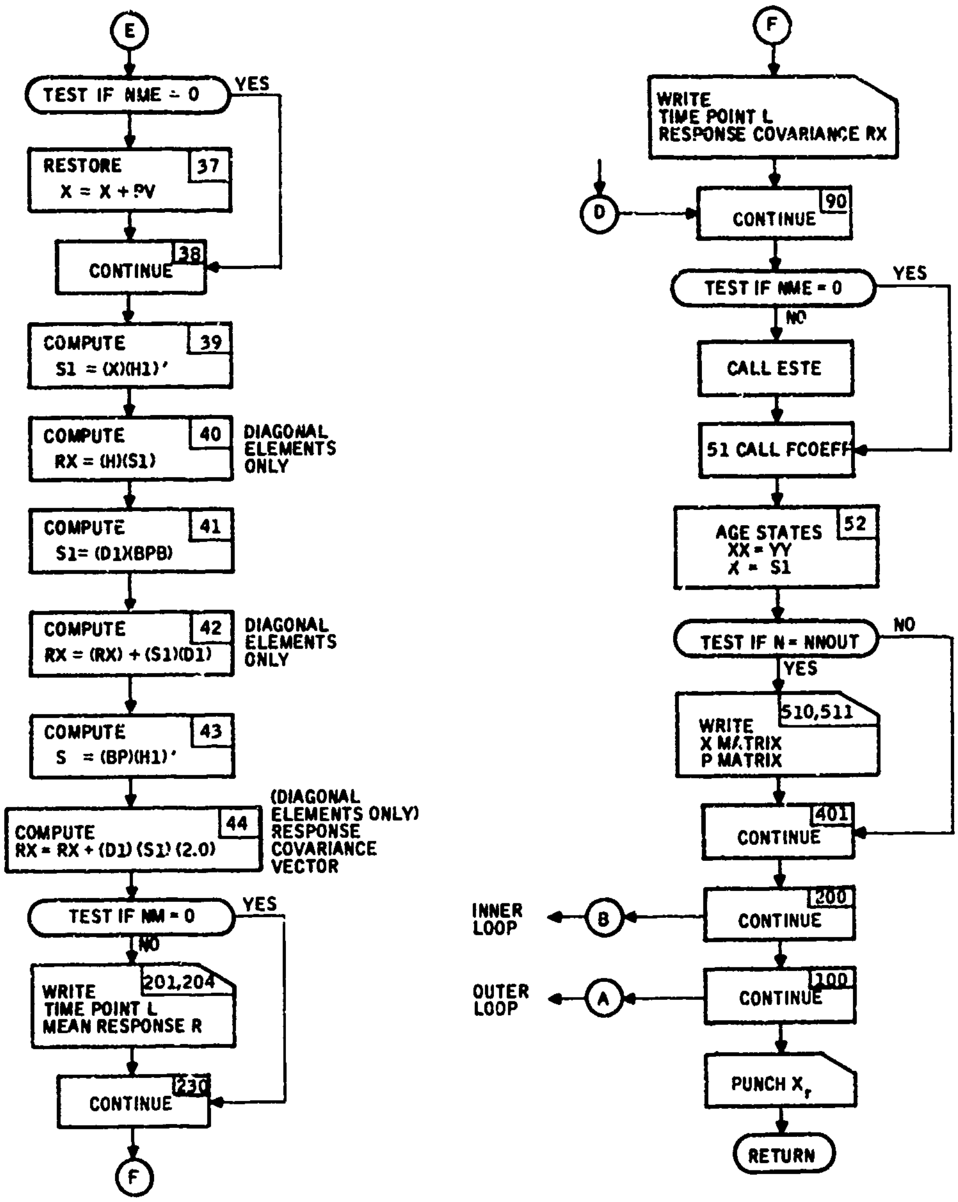


Figure 63. Subroutine COV Flow Diagram (concluded)

```
      SUBROUTINE COV
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON QD(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,[illegible]),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,D[illegible]V(4),DVW(3)
      REAL KV
      REWIND ITAPE
      REWIND NTAPE
      LP=[illegible]
      RD[illegible]
C     INITIALIZE ARRAYS
      M[illegible]
      D[illegible]=1,NX
      X[illegible]=0.
      YY(I)=0.
      DO 7 J=1,NX
      X(I,J)=0.
    7 PV(I,J)=0.
    1 CONTINUE
      CALL INPT(X,17,17)
      NX1=NX+1
      IF(NME)501,500,501
  501 DO 502 I=1,NX
      DO 502 J=1,NX
  502 PV(I,J)=X(I,J)
  500 CONTINUE
C
C READ INITIAL DATA AND FIRST DIFFERENCE
C
      READ(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      READ(ITAPE)((B3(I,J),J=1,NW),I=1,NX)
      READ(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      READ(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      READ(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      READ(ITAPE)VW
      READ(NTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      DO 333 I=1,NU
  333 FV(I)=KV(I,NX1)
      DO 4 I=1,NX
      AD(I)=A(I,I)
    4 A(I,I)=AD(I)+1.
      DO 100 L=1,MO
      NNOUT=NFREQ
      READ(ITAPE)((DA(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((DB1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((DB2(I,J),J=1,3),I=1,NX)
      READ(ITAPE)((DB3(I,J),J=1,NW),I=1,NX)
```

Figure 64. Subroutine COV Program Listing

```
      READ(ITAPE)((DH1(I,J),J=1,NX),I=1,NRE)
      READ(ITAPE)((DD1(I,J),J=1,NU),I=1,NRE)
      READ(ITAPE)((DD2(I,J),J=1,3),I=1,NRE)
      READ(ITAPE)DVW
      READ(NTAPE)((DKV(I,J),J=1,NX1),I=1,NU)
      DO 3 I=1,NU
    3 DFV(I)=DKV(I,NX1)
      IF(L.EQ.MO) GOTO 5
      MI=MIR
      GOTO 6
    5 MI=MIL
      NNOUT=MI
    6 CONTINUE
      DO 200 N=1,MI
C
C COMPUTE AXA +B3WB3/DT
C
      DO 10 I=1,NX
      DO 10 J=1,NX
      S1(I,J)=0.
      DO 10 K=1,NX
   10 S1(I,J)=S1(I,J)+X(I,K)*A(J,K)
      DO 11 I=1,NX
      DO 11 J=I,NX
      S2(I,J)=0.
      DO 11 K=1,NX
   11 S2(I,J)=S2(I,J)+A(I,K)*S1(K,J)
      DO 12 I=1,NW
      DO 12 J=1,NX
      S3(I,J)=0.
      DO 12 K=1,NW
   12 S3(I,J)=S3(I,J)+W1(I,K)*B3(J,K)
      DO 13 I=1,NX
      DO 13 J=I,NX
      DO 13 K=1,NW
   13 S2(I,J)=S2(I,J)+B3(I,K)*S3(K,J)*RDT
      IF(NM)14,21,14
   14 CONTINUE
C
C UPDATE MEAN STATE
C
      DO 15 I=1,NX
      YY(I)=B2(I,1)*VW(1)+B2(I,2)*VW(2)+B2(I,3)*VW(3)
      DO 15 J=1,NX
   15 YY(I)=YY(I)+A(I,J)*XX(J)
      DO 16 I=1,NU
      S1(I,1)=FV(I)
      DO 16 J=1,NX
   16 S1(I,1)=S1(I,1)+KV(I,J)*XX(J)
      DO 17 I=1,NX
      DO 17 J=1,NU
   17 YY(I)=YY(I)+B1(I,J)*S1(J,1)
C FORM X-P
   21 IF(NME)18,20,18
```

Figure 64. Subroutine COV Program Listing (continued)

```
   18 DO 19 I=1,NX
      DO 19 J=1,NX
   19 X(I,J)=X(I,J)-PV(I,J)
   20 CONTINUE
C
C FINISH UPDATING OF X IF B1KXA,AXKB1,AND B1KXKB1
C
      DO 22 I=1,NU
      DO 22 J=1,NX
      BP(I,J)=0.
      DO 22 K=1,NX
   22 BP(I,J)=BP(I,J)+KV(I,K)*X(K,J)
      DO 23 I=1,NU
      DO 23 J=1,NU
      BPB(I,J)=0.
      DO 23 K=1,NX
   23 BPB(I,J)=BPB(I,J)+BP(I,K)*KV(J,K)
      DO 24 I=1,NU
      DO 24 J=1,NX
      S3(I,J)=0.
      DO 24 K=1,NU
   24 S3(I,J)=S3(I,J)+BPB(I,K)*B1(J,K)
      DO 25 I=1,NX
      DO 25 J=I,NX
      DO 25 K=1,NU
   25 S2(I,J)=S2(I,J)+B1(I,K)*S3(K,J)
      DO 26 I=1,NU
      DO 26 J=1,NX
      S3(I,J)=0.
      DO 26 K=1,NX
   26 S3(I,J)=S3(I,J)+BP(I,K)*A(J,K)
      DO 27 I=1,NX
      DO 27 J=1,NX
      S1(I,J)=0.
      DO 27 K=1,NU
   27 S1(I,J)=S1(I,J)+B1(I,K)*S3(K,J)
      DO 228 I=1,NX
      DO 228 J=1,NX
  228 S3(J,I)=S1(I,J)
      DO 28 I=1,NX
      DO 28 J=I,NX
   28 S2(I,J)=S2(I,J)+S1(I,J)+S3(I,J)
      IF(NNOUT-N)90,30,90
   30 CONTINUE
      IF(NM)31,35,31
C COMPUTE MEAN RESPONSE
C
   31 CONTINUE
      DO 32 I=1,NRE
      R(I,1)=D2(I,1)*VW(1)+D2(I,2)*VW(2)+D2(I,3)*VW(3)
      DO 32 J=1,NX
   32 R(I,1)=R(I,1)+H1(I,J)*XX(J)
      DO 33 I=1,NU
      S1(I,1)=FV(I)
```

Figure 64. Subroutine COV Program Listing (continued)

```
      DO 33 J=1,NX
   33 S1(I,1)=S1(I,1)+KV(I,J)*XX(J)
      DO 34 I=1,NRE
      DO 34 J=1,NU
   34 R(I,1)=R(I,1)+D1(I,J)*S1(J,1)
   35 CONTINUE
C
C COMPUTE  RESPONSE COVARIANCES
C
      IF(NME)36,38,36
   36 DO 37 I=1,NX
      DO 37 J=1,NX
   37 X(I,J)=X(I,J)+PV(I,J)
   38 CONTINUE
      DO 39 I=1,NX
      DO 39 J=1,NRE
      S1(I,J)=0.
      DO 39 K=1,NX
   39 S1(I,J)=S1(I,J)+X(I,K)*H1(J,K)
      DO 40 I=1,NRE
      RX(I,1)=0.
      DO 40 J=1,NX
   40 RX(I,1)=RX(I,1)+H1(I,J)*S1(J,I)
      DO 41 I=1,NRE
      DO 41 J=1,NU
      S1(I,J)=0.
      DO 41 K=1,NU
   41 S1(I,J)=S1(I,J)+D1(I,K)*BPB(K,J)
      DO 42 I=1,NRE
      DO 42 J=1,NU
   42 RX(I,1)=RX(I,1)+S1(I,J)*D1(I,J)
      DO 43 I=1,NU
      DO 43 J=1,NRE
      S1(I,J)=0.
      DO 43 K=1,NX
   43 S1(I,J)=S1(I,J)+BP(I,K)*H1(J,K)
      DO 44 I=1,NRE
      DO 44 J=1,NU
   44 RX(I,1)=RX(I,1)+D1(I,J)*S1(J,I)*2.
      IF(NM)220,230,220
  220 WRITE(LW,201)
  201 FORMAT(1H1/7X,15H MEAN RESPONSES//)
      I=1
      WRITE(LW,204)L
      WRITE(LW,203)(J,R(J,I),J=1,NRE)
  203 FORMAT(8(2H RI2,1H=E10.3))
  204 FORMAT(12H TIME POINT I2//)
  202 CONTINUE
  230 CONTINUE
      WRITE(LW,205)
  205 FORMAT(1H1/7X,21H RESPONSE COVARIANCES//)
      I=1
      WRITE(LW,204)L
      WRITE(LW,211)(J,RX(J,I),J=1,NRE)
  211 FORMAT(8(2H SI2,1H=E10.3))
  210 CONTINUE
   90 CONTINUE
      IF(NME)50,51,50
   50 CALL FSTE(N,NNOUT)
   51 CALL FCOEF(L)
      DO 52 I=1,NX
      XX(I)=YY(I)
      DO 52 J=I,NX
      X(I,J)=S2(I,J)
   52 X(J,I)=S2(I,J)
      IF(N-NNOUT)401,400,401
  400 NNOUT=NNOUT+NFREQ
      WRITE(LW,510)
  510 FORMAT(1H1/7X,10H X MATRIX /)
      CALL MP(17,17,NX,NX,X,LW)
  401 CONTINUE
  200 CONTINUE
  100 CONTINUE
      CALL OUTP(17,17,NX,NX,X,LP)
      RETURN
      END
```

Figure 64. Subroutine COV Program Listing (concluded)

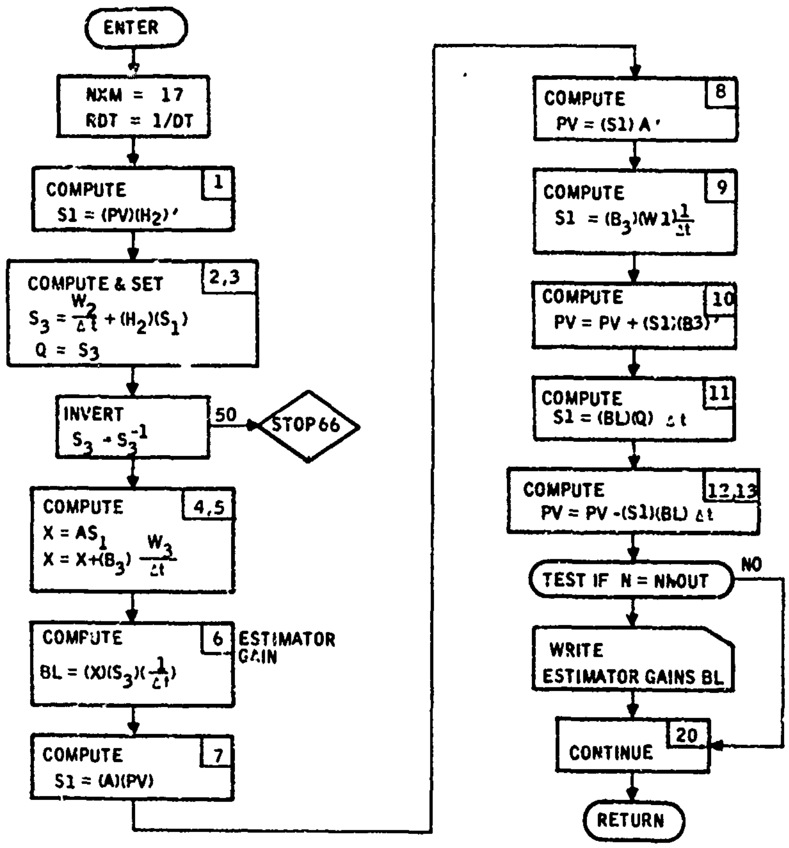


Figure 65. Subroutine ESTE Flow Diagram

```
      SUBROUTINE ESTF(N,NNOUT)
      DIMENSION KWAA(17)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      NXM=17
      RDT=1./DT
      DO 1 I=1,NX
      DO 1 J=1,NRT
      S1(I,J)=0.
      DO 1 K=1,NX
    1 S1(I,J)=S1(I,J)+PV(I,K)*H2(J,K)
      DO 2 I=1,NRT
      DO 2 J=1,NRT
      S3(I,J)=W2(I,J)*RDT
      DO 3 K=1,NX
    3 S3(I,J)=S3(I,J)+H2(I,K)*S1(K,J)
    2 Q(I,J)=S3(I,J)
      CALL TDINVR(ISOL,IDSOL,NRT,NRT,S3, 21,KWAA,DET)
      IF((ISOL+IDSOL).GT.2) GOTO 50
      GOTO 51
   50 STOP 66
   51 CONTINUE
      DO 4 I=1,NX
      DO 4 J=1,NRT
      X(I,J)=0.
      DO 5 K=1,NX
    5 X(I,J)=X(I,J)+A(I,K)*S1(K,J)
      DO 4 K=1,NW
    4 X(I,J)=X(I,J)+B3(I,K)*W3(K,J)*RDT
      DO 6 I=1,NX
      DO 6 J=1,NRT
      BL(I,J)=0.
      DO 6 K=1,NRT
    6 BL(I,J)=BL(I,J)+X(I,K)*S3(K,J)*RDT
      DO 7 I=1,NX
      DO 7 J=1,NX
      S1(I,J)=0.
      DO 7 K=1,NX
    7 S1(I,J)=S1(I,J)+A(I,K)*PV(K,J)
      DO 8 I=1,NX
      DO 8 J=I,NX
      PV(I,J)=0.
      DO 8 K=1,NX
    8 PV(I,J)=PV(I,J)+S1(I,K)*A(J,K)
      DO 9 I=1,NX
      DO 9 J=1,NW
      S1(I,J)=0.
      DO 9 K=1,NW
    9 S1(I,J)=S1(I,J)+B3(I,K)*W1(K,J)*RDT
      DO 10 I=1,NX
      DO 10 J=I,NX
      DO 10 K=1,NW
   10 PV(I,J)=PV(I,J)+S1(I,K)*B3(J,K)
      DO 11 I=1,NX
      DO 11 J=1,NRT
      S1(I,J)=0.
      DO 11 K=1,NRT
   11 S1(I,J)=S1(I,J)+BL(I,K)*Q(K,J)*DT
      DO 12 I=1,NX
      DO 12 J=I,NX
      DO 13 K=1,NRT
   13 PV(I,J)=PV(I,J)-S1(I,K)*BL(J,K)*DT
   12 PV(J,I)=PV(I,J)
      IF(N-NNOUT)20,21,20
   21 CONTINUE
      WRITE(LW,22)
   22 FORMAT(1H1/7X,16H ESTIMATOR GAINS//)
      CALL MP(17,12,NX,NRT,BL,LW)
      WRITE(LW,511)
  511 FORMAT(1H1/7X,10H PH MATRIX/)
      CALL MP(17,17,NX,NX,PV,LW)
   20 CONTINUE
      RETURN
      END
```

Figure 66. Subroutine ESTE Program Listing

ADAP 2 DATA MANIPULATION SUBROUTINES

Subroutine DATAGEN

Subroutine DATAGEN implements the analysis given in Section VI, Volume I. It generates the shuffled and augmented linear data. It reads the state component shuffling data and the original linear data stored in permanent disc file by ADAP 1. It shuffles the data in accordance with the shuffling indices and writes the augmented data in a scratch disc file. Its flow chart is shown in Figure 67 and its program listing in Figure 68.

Subroutine SHUF

Subroutine SHUF shuffles the matrices in accordance with the shuffling indices. It is called by subroutine DATAGEN. The subroutines flow chart is shown in Figure 69 and the program listing in Figure 70.

Subroutine REVS

Subroutine REVS reads the shuffled and augmented linear data from scratch tape and reverses their order with respect to time points using the relation

$$\phi(t_k) = f(t_n - t_k)$$

It stores this data into scratch tape. The backward time data is used by subroutine COV.

The flow diagram of subroutine REVS is shown in Figure 71 and the program listing in Figure 72.

Subroutine DIFBC

Subroutine DIFBC reads in the linear data and computes the current differences from

$$\Delta f(\ell) = [f(\ell - 1) - f(\ell)]\left(\frac{\Delta t}{\Delta T}\right)$$

and writes on a scratch tape. The data updating uses this difference.

The subroutines flow chart is shown in Figure 73 and the program listing in Figure 74.

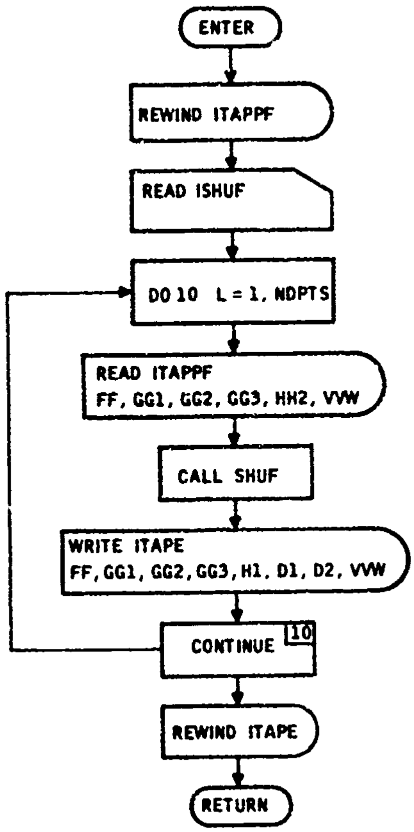


Figure 67. Subroutine DATAGEN Flow Diagram

```
      SUBROUTINE DATAGEN(ITAPPF,NDPTS)
      DIMENSION FF(20,20),GG1(20,8),GG2(20,6),GG3(20,4),HH2(21,12),VVW(3
     1),ISHUF(42)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REWIND ITAPPF
      READ(LR,11)ISHUF
   11 FORMAT(21I2)
      DO 10 L=1,NDPTS
      READ(ITAPPF)FF
      READ(ITAPPF)GG1
      READ(ITAPPF)GG2
      READ(ITAPPF)GG3
      READ(ITAPPF)HH2
      READ(ITAPPF)VVW
      CALL SHUF(FF,GG1,GG2,GG3,VVW,ISHUF)
      WRITE(ITAPE)((FF(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPE)((GG1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((GG2(I,J),J=1, 3),I=1,NX)
      WRITE(ITAPE)((GG3(I,J),J=1,NW),I=1,NX)
      WRITE(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(ITAPE)VVW
   10 CONTINUE
      REWIND ITAPE
      RETURN
      END
```

Figure 68. Subroutine DATAGEN Program Listing

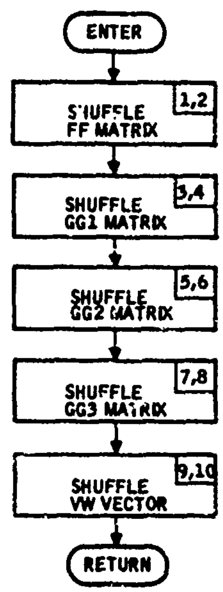


Figure 69. Subroutine SHUF Flow Diagram

```
      SUBROUTINE SHUF(FF,GG1,GG2,GG3,VW,ISHUF)
      DIMENSION FF(20,20),GG1(20,8),GG2(20,6),GG3(20,4),VW(3)
      DIMENSION D(20,20),ISHUF(42)
C
C  SHUFFLE F MATRIX
C
      DO 1 I=1,20
      II=ISHUF(I)
      DO 1 J=1,20
    1 D(I,J)=FF(II,J)
      DO 2 I=1,20
      DO 2 J=1,20
      JJ=ISHUF(J)
    2 FF(I,J)=D(I,JJ)
C
C  SHUFFLE G1 MATRIX
C
      DO 3 I=1,20
      II=ISHUF(I)
      DO 3 J=1,8
    3 D(I,J)=GG1(II,J)
      DO 4 I=1,20
      DO 4 J=1,8
      JJ=ISHUF(J+20)
    4 GG1(I,J)=D(I,JJ)
C
C  SHUFFLE G2 MATRIX
C
      DO 5 I=1,20
      II=ISHUF(I)
      DO 5 J=1,6
    5 D(I,J)=GG2(II,J)
      DO 6 I=1,20
      DO 6 J=1,6
      JJ=ISHUF(J+28)
    6 GG2(I,J)=D(I,JJ)
C
C  SHUFFLE G3 MATRIX
C
      DO 7 I=1,20
      II=ISHUF(I)
      DO 7 J=1,4
    7 D(I,J)=GG3(II,J)
      DO 8 I=1,20
      DO 8 J=1,4
      JJ=ISHUF(J+34)
    8 GG3(I,J)=D(I,JJ)
C
C  SHUFFLE VW
C
      DO 9 I=1,3
      II=ISHUF(I+38)
    9 D(I,1)=VW(II)
      DO 10 I=1,3
   10 VW(I)=D(I,1)
      RETURN
      END
```

Figure 70. Subroutine SHUF Program Listing

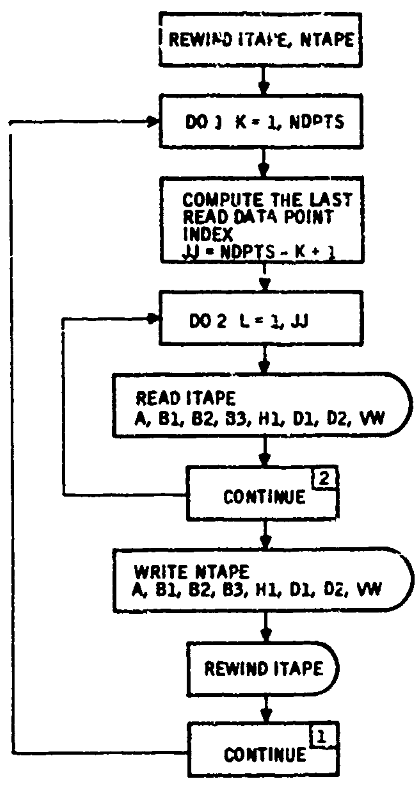


Figure 71. Subroutine REVS Flow Diagram

```
      SUBROUTINE REVS(NDPTS)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      REWIND ITAPE
      REWIND NTAPE
      DO 1 K=1,NDPTS
      JJ=NDPTS-K+1
      DO 2 L=1,JJ
      READ(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      READ(ITAPE)((B3(I,J),J=1,NW),I=1,NX)
      READ(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      READ(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      READ(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      READ(ITAPE)VW
    2 CONTINUE
      WRITE(NTAPE)((A(I,J),J=1,NX),I=1,NX)
      WRITE(NTAPE)((B1(I,J),J=1,NU),I=1,NX)
      WRITE(NTAPE)((B2(I,J),J=1,3),I=1,NX)
      WRITE(NTAPE)((B3(I,J),J=1,NW),I=1,NX)
      WRITE(NTAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(NTAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(NTAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(NTAPE)VW
      REWIND ITAPE
    1 CONTINUE
      REWIND NTAPE
      RETURN
      END
```

Figure 72. Subroutine REVS Program Listing

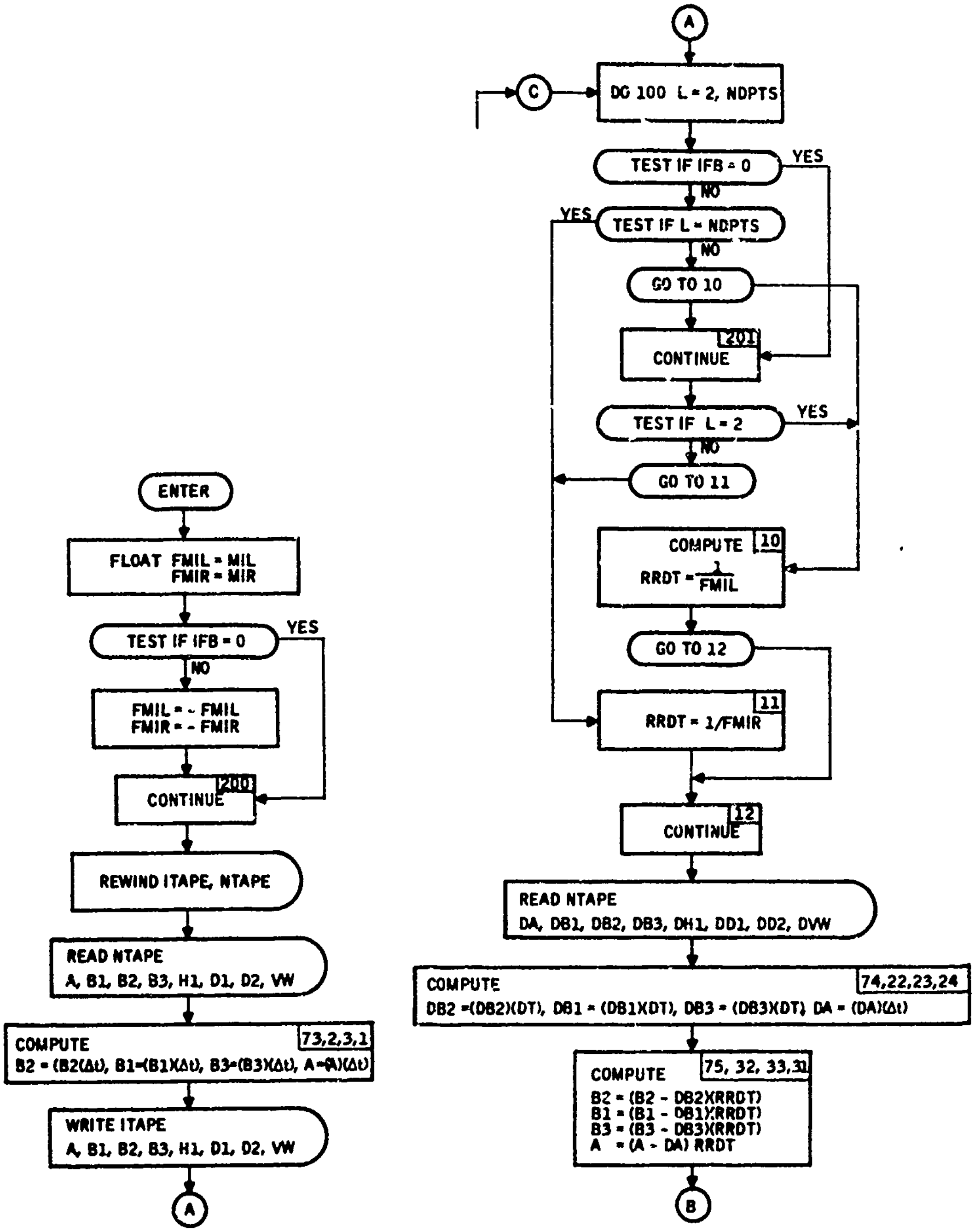


Figure 73. Subroutine DIFBC Flow Diagram

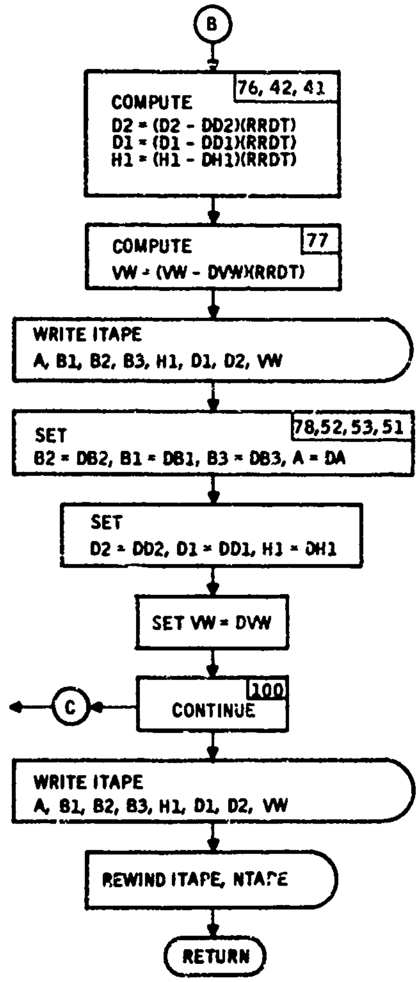


Figure 73. Subroutine DIFBC Flow Diagram (concluded)

```
      SUBROUTINE DIFBC(NDPTS,IFB)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      FMIL=MIL
      FMIR=MIR
      IF(IFB.EQ.0) GOTO 200
      FMIL=-FMIL
      FMIR=-FMIR
  200 CONTINUE
      REWIND ITAPE
      REWIND NTAPE
      READ(NTAPE)((A(I,J),J=1,NX),I=1,NX)
      READ(NTAPE)((B1(I,J),J=1,NU),I=1,NX)
      READ(NTAPE)((B2(I,J),J=1,3),I=1,NX)
      READ(NTAPE)((B3(I,J),J=1,NW),I=1,NX)
      READ(NTAPE)((H1(I,J),J=1,NX),I=1,NRE)
      READ(NTAPE)((D1(I,J),J=1,NU),I=1,NRE)
      READ(NTAPE)((D2(I,J),J=1,3),I=1,NRE)
      READ(NTAPE)VW
      DO 1 I=1,NX
      DO 73 J=1,3
   73 B2(I,J)=B2(I,J)*DT
      DO 2 J=1,NU
    2 B1(I,J)=B1(I,J)*DT
      DO 3 J=1,NW
    3 B3(I,J)=B3(I,J)*DT
      DO 1 J=1,NX
    1 A(I,J)=A(I,J)*DT
      WRITE(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      WRITE(ITAPE)((B3(I,J),J=1,NW),I=1,NX)
      WRITE(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(ITAPE)VW
      DO 100 L=2,NDPTS
      IF(IFB.EQ.0) GOTO 201
      IF(L.EQ.NDPTS) GOTO 11
      GOTO 10
  201 CONTINUE
      IF(L.EQ.2) GOTO 10
      GOTO 11
   10 RRDT=1./FMIL
      GOTO 12
   11 RRDT=1./FMIR
```

Figure 74. Subroutine DIFBC Program Listing

```
   12 CONTINUE
      READ(NTAPE)((DA(I,J),J=1,NX),I=1,NX)
      READ(NTAPE)((DB1(I,J),J=1,NU),I=1,NX)
      READ(NTAPE)((DB2(I,J),J=1,3),I=1,NX)
      READ(NTAPE)((DB3(I,J),J=1,NW),I=1,NX)
      READ(NTAPE)((DH1(I,J),J=1,NX),I=1,NRE)
      READ(NTAPE)((DD1(I,J),J=1,NU),I=1,NRE)
      READ(NTAPE)((DD2(I,J),J=1,3),I=1,NRE)
      READ(NTAPE)DVW
      DO 21 I=1,NX
      DO 74 J=1,3
   74 DB2(I,J)=DB2(I,J)*DT
      DO 22 J=1,NU
   22 DB1(I,J)=DB1(I,J)*DT
      DO 23 J=1,NW
   23 DB3(I,J)=DB3(I,J)*DT
      DO 21 J=1,NX
   21 DA(I,J)=DA(I,J)*DT
      DO 31 I=1,NX
      DO 75 J=1,3
   75 B2(I,J)=(B2(I,J)-DB2(I,J))*RRDT
      DO 32 J=1,NU
   32 B1(I,J)=(B1(I,J)-DB1(I,J))*RRDT
      DO 33 J=1,NW
   33 B3(I,J)=(B3(I,J)-DB3(I,J))*RRDT
      DO 31 J=1,NX
   31 A(I,J)=(A(I,J)-DA(I,J))*RRDT
      DO 41 I=1,NRE
      DO 76 J=1,3
   76 D2(I,J)=(D2(I,J)-DD2(I,J))*RRDT
      DO 42 J=1,NU
   42 D1(I,J)=(D1(I,J)-DD1(I,J))*RRDT
      DO 41 J=1,NX
   41 H1(I,J)=(H1(I,J)-DH1(I,J))*RRDT
      DO 77 I=1,3
   77 VW(I)=(VW(I)-DVW(I))*RRDT
      WRITE(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      WRITE(ITAPE)((B3(I,J),J=1,NW),I=1,NX)
      WRITE(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(ITAPE)VW
      DO 51 I=1,NX
      DO 78 J=1,3
   78 B2(I,J)=DB2(I,J)
      DO 52 J=1,NU
   52 B1(I,J)=DB1(I,J)
      DO 53 J=1,NW
   53 B3(I,J)=DB3(I,J)
      DO 51 J=1,NX
   51 A(I,J)=DA(I,J)
      DO 61 I=1,NRE
      DO 79 J=1,3
   79 D2(I,J)=DD2(I,J)
      DO 62 J=1,NU
   62 D1(I,J)=DD1(I,J)
      DO 61 J=1,NX
   61 H1(I,J)=DH1(I,J)
      DO 80 I=1,3
   80 VW(I)=DVW(I)
  100 CONTINUE
      WRITE(ITAPE)((A(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPE)((B1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((B2(I,J),J=1,3),I=1,NX)
      WRITE(ITAPE)((B3(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(ITAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(ITAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(ITAPE)VW
      REWIND ITAPE
      REWIND NTAPE
      RETURN
      END
```

Figure 74. Subroutine DIFBC Program Listing (concluded)

Subroutine BCOEF

Subroutine BCOEF generates the current value of linear data backward in time from

$$f(\ell) = f(\ell - 1) - \Delta f(\ell)$$

where $\Delta f(\ell)$ is computed by the subroutine DIFBC.

The flow chart of the subroutine BCOEF is shown in Figure 75 and the program listing in Figure 76.

Subroutine DIFG

Subroutine DIFG generates the current slope of controller gains from

$$\Delta K(\ell) = [K(\ell - 1) - K(\ell)] / \Delta T$$

and stores them in a scratch tape. It basically performs the same function as the subroutine DIFBC.

The flow chart of subroutine DIFG is shown in Figure 77 and the program listing in Figure 78.

Subroutine RDWT

Subroutine RDWT transfers the linear data from one scratch tape to another. Its flow chart is shown in Figure 79 and its program listing in Figure 80.

Subroutine REVG

Subroutine REVG basically performs the same function as subroutine REVS except it reverses in time the current slopes of controller gains using

$$K(t_k) = K(t_n - t_k)$$

The flow chart of subroutine REVG is shown in Figure 81 and its program listing in Figure 82.

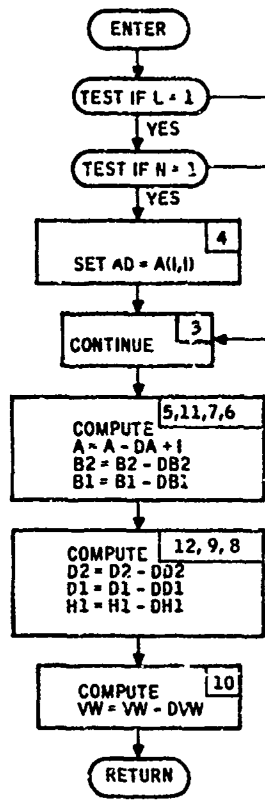


Figure 75. Subroutine BCOEF Flow Diagram

```
      SUBROUTINE BCOFF(L,N)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      IF(L-1)3,1,3
    1 IF(N-1)3,2,3
    2 DO 4 I=1,NX
    4 AD(I)=A(I,I)
    3 CONTINUE
      DO 5 I=1,NX
      AD(I)=AD(I)-DA(I,I)
    5 A(I,I)=AD(I)+1.
      DO 6 I=1,NX
      DO 11 J=1,3
   11 B2(I,J)=B2(I,J)-DB2(I,J)
      DO 7 J=1,NU
    7 B1(I,J)=B1(I,J)-DB1(I,J)
      DO 6 J=1,NX
      IF(I.EQ.J) GOTO 6
      A(I,J)=A(I,J)-DA(I,J)
    6 CONTINUE
      DO 8 I=1,NR
      DO 12 J=1,3
   12 D2(I,J)=D2(I,J)-DD2(I,J)
      DO 9 J=1,NU
    9 D1(I,J)=D1(I,J)-DD1(I,J)
      DO 8 J=1,NX
    8 H1(I,J)=H1(I,J)-DH1(I,J)
      DO 10 I=1,3
   10 VW(I)=VW(I)-DVW(I)
      RETURN
      END
```

Figure 76. Subroutine BCOEF Program Listing

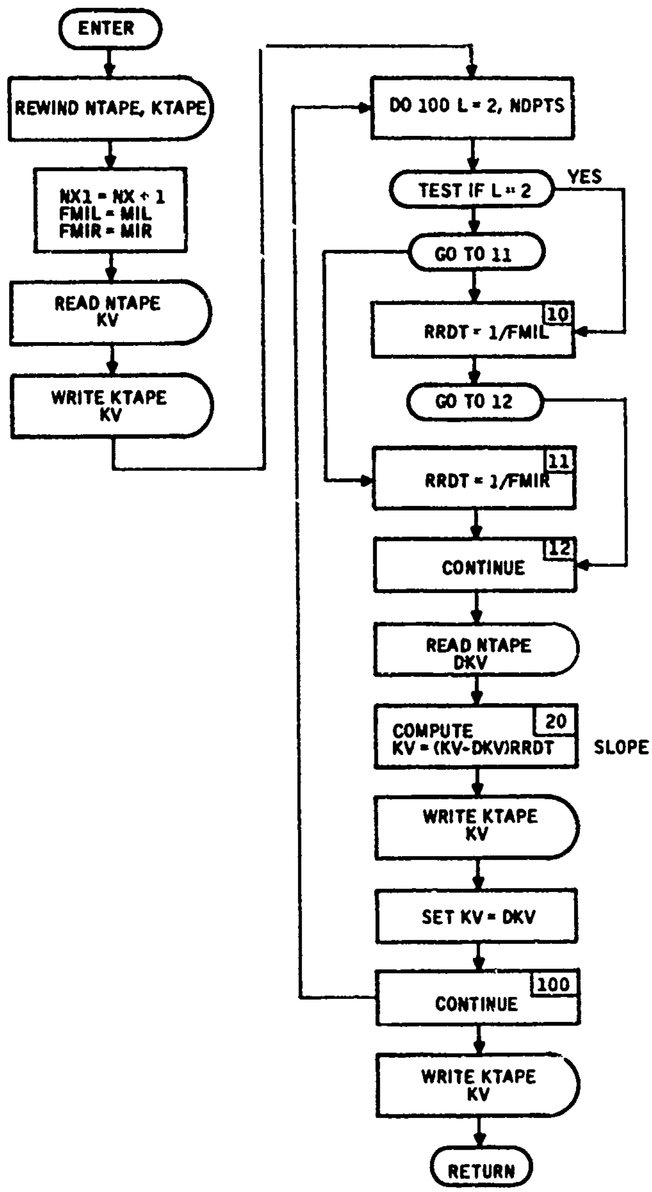


Figure 77. Subroutine DIFG Flow Diagram

```
      SUBROUTINE DIFG(KTAPE,NDPTS)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      REWIND NTAPE
      REWIND KTAPE
      NX1=NX+1
      FMIL=MIL
      FMIR=MIR
      READ(NTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      WRITE(KTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      DO 100 L=2,NDPTS
      IF(L.EQ.2) GOTO 10
      GOTO 11
   10 RRDT=1./FMIL
      GOTO 12
   11 RRDT=1./FMIR
   12 CONTINUE
      READ(NTAPE)((DKV(I,J),J=1,NX1),I=1,NU)
      DO 20 I=1,NU
      DO 20 J=1,NX1
   20 KV(I,J)=(KV(I,J)-DKV(I,J))*RRDT
      WRITE(KTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      DO 21 I=1,NU
      DO 21 J=1,NX1
   21 KV(I,J)=DKV(I,J)
  100 CONTINUE
      WRITE(KTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      RETURN
      END
```

Figure 78. Subroutine DIFG Program Listing

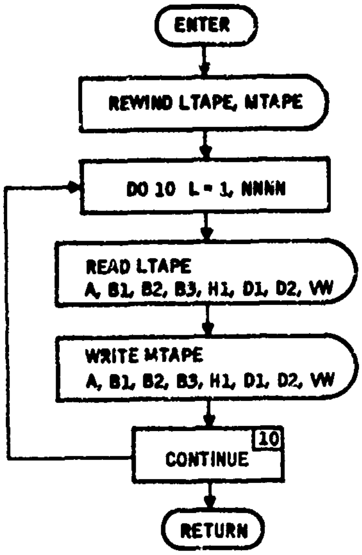


Figure 79. Subroutine RDWT Flow Diagram

```
      SUBROUTINE RDWT(LTAPE,MTAPE,NNNN)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRF,NRB,DFV(4),DVW(3)
      REAL KV
      REWIND LTAPE
      REWIND MTAPE
      DO 10 L=1,NNNN
      READ(LTAPE)((A(I,J),J=1,NX),I=1,NX)
      READ(LTAPE)((B1(I,J),J=1,NU),I=1,NX)
      READ(LTAPE)((B2(I,J),J=1,3),I=1,NX)
      READ(LTAPE)((B3(I,J),J=1,NW),I=1,NX)
      READ(LTAPE)((H1(I,J),J=1,NX),I=1,NRE)
      READ(LTAPE)((D1(I,J),J=1,NU),I=1,NRE)
      READ(LTAPE)((D2(I,J),J=1,3),I=1,NRE)
      READ(LTAPE)VW
      WRITE(MTAPE)((A(I,J),J=1,NX),I=1,NX)
      WRITE(MTAPE)((B1(I,J),J=1,NU),I=1,NX)
      WRITE(MTAPE)((B2(I,J),J=1,3),I=1,NX)
      WRITE(MTAPE)((B3(I,J),J=1,NW),I=1,NX)
      WRITE(MTAPE)((H1(I,J),J=1,NX),I=1,NRE)
      WRITE(MTAPE)((D1(I,J),J=1,NU),I=1,NRE)
      WRITE(MTAPE)((D2(I,J),J=1,3),I=1,NRE)
      WRITE(MTAPE)VW
   10 CONTINUE
      RETURN
      END
```

Figure 80. Subroutine RDWT Program Listing

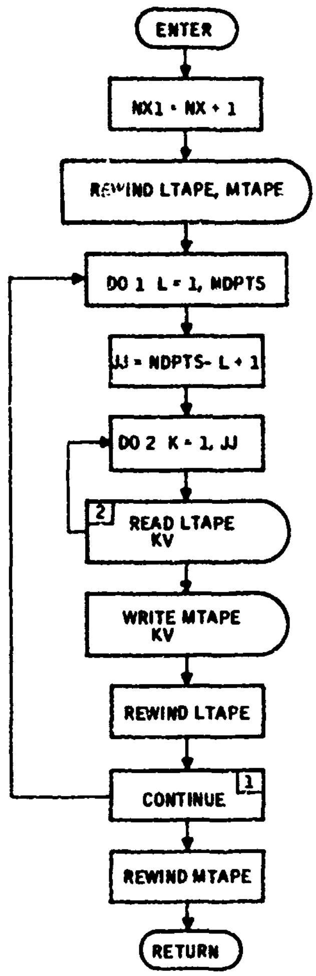


Figure 81. Subroutine REVG Flow Diagram

```
      SUBROUTINE REVG(LTAPE,MTAPE,NDPTS)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DQD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3)  ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,12)
      COMMON NX,NK,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      NX1=NX+1
      REWIND LTAPE
      REWIND MTAPE
      DO 1 L=1,NDPTS
      JJ=NDPTS-L+1
      DO 2 K=1,JJ
    2 READ(LTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      WRITE(MTAPE)((KV(I,J),J=1,NX1),I=1,NU)
      REWIND LTAPE
    1 CONTINUE
      REWIND MTAPE
      RETURN
      END
```

Figure 82. Subroutine REVG Program Listing

Subroutine FCOEF

Subroutine FCOEF generates the current value of the forward time linear data from

$$f(k) = f(k - 1) + \Delta f(k)$$

The flow diagram of subroutine FCOEF is shown in Figure 83 and the program listing in Figure 84.

ADAP 2 AUXILIARY SUBROUTINES

Subroutine MP

Subroutine MP prints the matrix quantities. Each row of printed matrix is identified. Its flow chart is shown in Figure 85 and its program listing in Figure 86.

Subroutine INPT

Subroutine INPT reads in matrices from cards. Each nonzero element of a matrix is identified by its row and column indices. Up to five elements can be input on one card.

The flow chart of subroutine INPT is shown in Figure 87 and the program listing in Figure 88.

Subroutine OUTP

Subroutine OUTP punches matrices into cards. Its flow chart is shown in Figure 89 and its program listing in Figure 90.

Subroutine TDINVR

Subroutine TDINVR is a general-purpose matrix inversion subroutine. It uses the Gaussian reduction. Its program listing is shown in Figure 91.

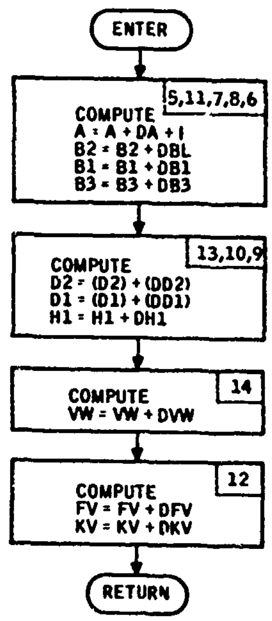


Figure 83. Subroutine FCOEF Flow Diagram

```
      SUBROUTINE FCOEF(L)
      COMMON A(17,17),DA(17,17),B1(17,4),DB1(17,4),B2(17,3),DB2(17,3)
      COMMON BPB(4,4),DUD(4,4),H1(21,17),DH1(21,17),D1(21,4),DD1(21,4)
      COMMON D2(21,3),DD2(21,3),PV(17,17),KV(4,18),FV(4),GN(17),AD(17)
      COMMON VW(3) ,S1(21,21),S2(21,21),S3(21,21),BP(4,17),DQ(4,21)
      COMMON SUM(4,4),KWA(4),Q(21,21),B3(17,3),DB3(17,3),DKV(4,18)
      COMMON NX,NR,NU,NW,NRT,NM,LW,LR,MO,MI,DT,MIL,MIR,NFREQ,ITAPE,NTAPE
      COMMON XX(17),YY(17),X(17,17),RX(21,1),R(21,1),W1(3,3),W2(12,12)
      COMMON W3(3,12),H2(12,17),BL(17,12)
      COMMON NME,NRE,NRB,DFV(4),DVW(3)
      REAL KV
      DO 5 I=1,NX
      AD(I)=AD(I)+DA(I,I)
    5 A(I,I)=AD(I)+1.
      DO 6 I=1,NX
      DO 11 J=1,3
   11 B2(I,J)=B2(I,J)+DB2(I,J)
      DO 7 J=1,NU
    7 B1(I,J)=B1(I,J)+DB1(I,J)
      DO 8 J=1,NW
    8 B3(I,J)=B3(I,J)+DB3(I,J)
      DO 6 J=1,NX
      IF(I.EQ.J) GOTO 6
      A(I,J)=A(I,J)+DA(I,J)
    6 CONTINUE
      DO 9 I=1,NR
      DO 13 J=1,3
   13 D2(I,J)=D2(I,J)+DD2(I,J)
      DO 10 J=1,NU
   10 D1(I,J)=D1(I,J)+DD1(I,J)
      DO 9  J=1,NX
    9 H1(I,J)=H1(I,J)+DH1(I,J)
      DO 14 I=1,3
   14 VW(I)=VW(I)+DVW(I)
      DO 12 I=1,NU
      FV(I)=FV(I)+DFV(I)
      DO 12 J=1,NX
   12 KV(I,J)=KV(I,J)+DKV(I,J)
      RETURN
      END
```

Figure 84. Subroutine FCOEF Program Listing

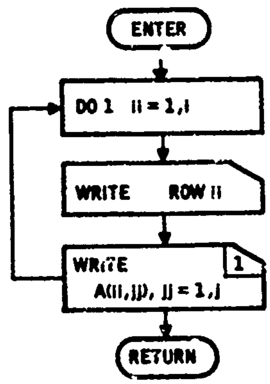


Figure 85. Subroutine MP Flow Diagram

```
      SUBROUTINE MP(K,L,I,J,A,LW)
      DIMENSION A(K,L)
      DO 1 II=1,I
      WRITE(LW,5)II
    5 FORMAT(5H ROW I3)
    1 WRITE(LW,2)(A(II,JJ),JJ=1,J)
    2 FORMAT(10E12.4)
      RETURN
      END
```

Figure 86. Subroutine MP Program Listing

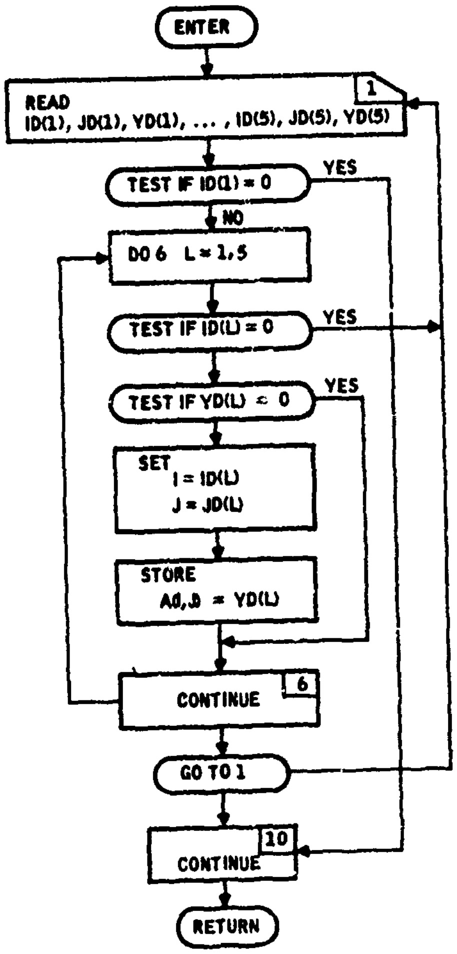


Figure 87. Subroutine INPT Flow Diagram

```
      SUBROUTINE INPT(A,II,JJ)
      DIMENSION A(II,JJ),ID(5),JD(5),YD(5)
    2 FORMAT(5(2I2,E12.3))
    1 READ(5,2)ID(1),JD(1),YD(1),ID(2),JD(2),YD(2),ID(3),JD(3),YD(3),
     1ID(4),JD(4),YD(4),ID(5),JD(5),YD(5)
      IF(ID(1))3,10,3
    3 DO 6 L=1,5
      IF(ID(L))4,1,4
    4 IF(YD(L))7,5,7
    7 I=ID(L)
      J=JD(L)
    5 A(I,J)=YD(L)
    6 CONTINUE
      GOTO 1
   10 CONTINUE
      RETURN
      END
```

Figure 88. Subroutine INPT Program Listing

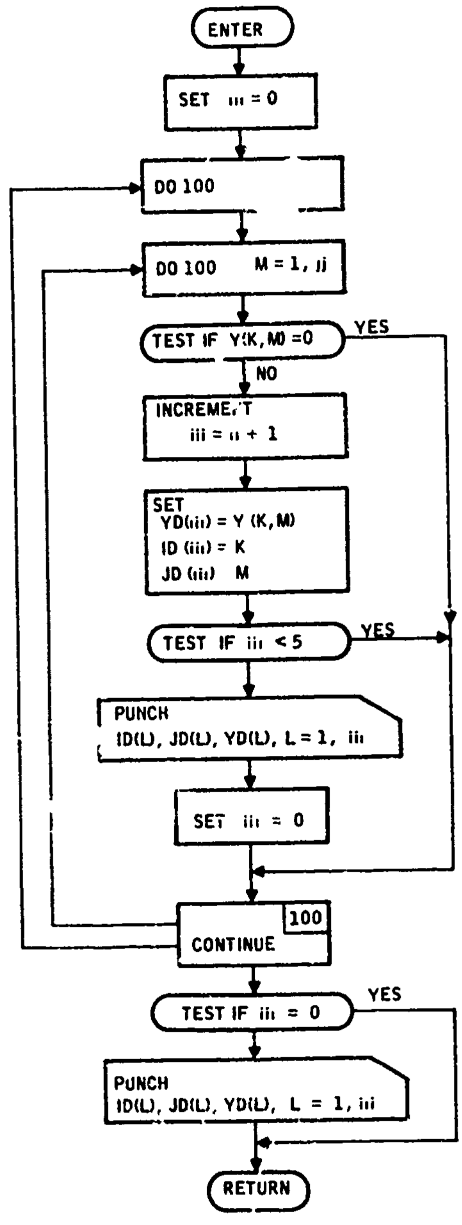


Figure 89. Subroutine OUTP Flow Diagram

```
      SUBROUTINE OUTP(I,J,II,JJ,Y,LP)
      DIMENSION Y(I,J),YD(5),ID(5),JD(5)
   50 FORMAT(5(2I3,F12.5))
      III=0
      DO 100 K=1,II
      DO 100 M=1,JJ
      IF(Y(K,M).EQ.0.) GOTO 100
      III=III+1
      YD(III)=Y(K,M)
      ID(III)=K
      JD(III)=M
      IF(III.LT.5) GOTO 100
      WRITE(LP,50)(ID(L),JD(L),YD(L),L=1,III)
      III=0
  100 CONTINUE
      IF(III.EQ.0) RETURN
      WRITE(LP,50)(ID(L),JD(L),YD(L),L=1,III)
      RETURN
      END
```

Figure 90. Subroutine OUTP Program Listing

```
      SUBROUTINE TDINVR(ISOL,IDSOL,NR,NC,A,MRA,KWA,DET)
      DIMENSION A(1),KWA(1)
      IR=NR
      ISOL =1
      IDSOL=1
      KK=2
   10 IF(NR)    61,61,11
   11 IF(IR-MRA)12,12,61
   12 IC=IABS(NC)
      IF(IC-IR) 13,14,14
   13 IC=IR
   14 IBMP=1
      JBMP=MRA
      KBMP=JBMP+IBMP
      NES=IR*JBMP
      NET=IC*JBMP
      IF(NC)  15,61,16
   15 MDIV=JBMP+1
      IRIC=IR-IC
      GO TO 17
   16 MDIV=1
   17 MAD =MDIV
      MSER=1
      KSER=IR
      MZ  =1
      DET=1.0
   18 PIV=0.0
      I=MSER
   19 IF(I-KSER)          20,20,23
   20 IF( ABS(A(I))-PIV)22,22,21
   21 PIV= ABS(A(I))
      IP=I
   22 I=I+IBMP
      GO TO 19
   23 IF(PIV)  24,62,24
   24 IF(NC)  26,25,25
   25 I=IP-((IP-1)/JBMP)*JBMP
      J=MSER-((MSER-1)/JBMP)*JBMP
      JJ=MSER/KBMP+1
      II=JJ+(IP-MSER)
      KWA(JJ)=II
      GO TO 27
   26 I=IP
      J=MSER
   27 IF(IP-MSER) 61,31,28
   28 IF(J-NET)  29,29,30
   29 PSTO=A(I)
      A(I)=A(J)
      A(J)=PSTO
      I=I+JBMP
      J=J+JBMP
      GO TO 28
   30 DET=-DET
   31 PSTO=A(MSER)
```

Figure 91. Subroutine TDINVR Program Listing

```
      KK=2
      GOTO(34,33),KK
   33 GO TO 35
   34 IDSOL=2
   35 PSTO=1.0/PSTO
      A(MSER)=1.0
      I=MDIV
   36 IF(I-NET)   37,37,39
   37 A(I)=A(I)*PSTO
      I=I+JBMP
      GO TO 36
   39 IF(MZ-KSER) 40,40,45
   40 IF(MZ-MSER) 41,44,41
   41 I=MAD
      J=MDIV
      PSTO=A(MZ)
      IF(PSTO) 142,44,142
  142 A(MZ)=0.0
   42 IF(J-NET)   43,43,44
   43 A(I)=A(I)-A(J)*PSTO
      J=J+JBMP
      I=I+JBMP
      GO TO 42
   44 MAD=MAD+IBMP
      MZ=MZ+IBMP
      GO TO 39
   45 KK=2
      GOTO(63,145),KK
  145 KSER=KSER+JBMP
      IF(KSER-NES) 46,46,53
   46 MSER=MSER+KBMP
      IF(NC) 48,47,47
   47 MDIV=MDIV+IBMP
      MZ=((MSER-1)/JBMP)*JBMP+1
      MAD=1
      GO TO 52
   48 MDIV=MDIV+KBMP
      IF(IRIC) 50,49,50
   49 MZ=MSER+IBMP
      GO TO 51
   50 MZ=((MSER-1)/JBMP)*JBMP+1
   51 MAD=MZ+JBMP
   52 GO TO 18
   53 IF(NC) 53,54,54
   54 JR=IR
   55 IF(JR) 61,65,56
   56 IF(KMA(JR)-JR) 61,60,57
   57 K=(JR-1)*JBMP
      J=K+IR
      L=(KMA(JR)-1)*JBMP+IR
   58 IF(J-K)   61,60,59
   59 PSTO=A(L)
      A(L)=A(J)
      A(J)=PSTO
      J=J-IBMP
      L=L-IBMP
      GO TO 58
   60 JR=JR-1
      GO TO 55
   61 ISOL=3
      GO TO 65
   62 DET=0.0
      ISOL=2
      IDSOL=1
      GO TO 65
   63 ISOL=2
      IDSOL=2
   65 RETURN
      END
```

Figure 91. Subroutine TDINVR Program Listing (concluded)

SECTION V

ADAP 3 (PERK) - NONSTATIONARY WEAPON PERFORMANCE PROGRAM

Program ADAP 3 implements the analysis developed in Section VIII of Volume I. It is essentially a time varying linear system simulation program. It develops perturbation trajectories for the mean and covariance of system state. In addition, it computes CEP performance measure, equivalent weights of the quadratic cost, and optimal control weighting matrix and the variance contribution matrix.

In this section input/output information is given first; then the main program and its subroutines are described.

ADAP 3 INPUT/OUTPUT

INPUT DESCRIPTION

Input for ADAP 3 is in the form of cards and data stored in a permanent disc file.

Card Data Input

The first group of cards to be read is cards 1-4 which provide basic program data. Their formats are shown in Table XXV.

The next cards in the data deck are the nonzero elements of the matrices X_r, H_B, H_r, and Σ_r. These data cards are read by the matrix input subroutines INPT. The subroutine INPT and the associated data card format is described in "Input ADAP 2 (DISCOP)."

Then the input cards for the vectors $\bar{x}_r$, f_r, and $\bar{\xi}_r$ are read. The format for the first card of a vector input is shown in Table XXVI.

The input card for the scalars $\bar{\delta t_r}$ and $\bar{\delta t_r^2}$ corresponding to the release-time error follows the last card of vector inputs. The format for this card is shown in Table XXVII.

The next two cards are for the state derivative components of the bomb at impact point. The format for these is shown in Table XXVIII.

The last portion of the input data deck is for the matrices X_f and ϕ, if IRUN $\neq$ 0

Figure 92 illustrates the sequence in the input card deck.

Table XXV. Format for ADAP 3 Data Input Cards 1-4

Card/Format	Column	Quantity	Description
1/(3I3)	1-3	N	Number of integration steps/second
	4-6	NX	Number of state variables
	709	NW	Number of disturbances
2/(2E15.8)	1-15	TR	Release time
	16-30	TF	Impact time of weapon
3/(20I2)	1-2	ISUF1	
	3-4	ISUF2	
	5-6	ISUF3	
	7-8	ISUF4	
	9-10	ISUF5	
	11-12	ISUF6	
	13-14	ISUF7	
	15-16	ISUF8	
	17-18	ISUF9	
	19-20	ISUF10	
	21-22	ISUF11	
	23-24	ISUF12	
	25-26	ISUF13	
	27-28	ISUF14	
	29-30	ISUF15	
	31-32	ISUF16	
	33-34	ISUF17	
	35-36	ISUF18	
	37-38	ISUF19	
	39-40	ISUF20	
4/(I2)	1-2	IRUN	IRUN = 0 Integrate to obtain X_f and ϕ IRUN $\neq$ 0 Read in X_f and ϕ

Table XXVI. Format for First Card of a Vector Input [Format (5E12.5)]

Column	Quantity	Description
1-12	V(1)	Value of the first component of a vector
13-24	V(2)	Value of the second component of a vector
25-36	V(3)	Value of the third component of a vector
37-48	V(4)	Value of the fourth component of a vector
49-60	V(5)	Value of the fifth component of a vector

Table XXVII. Format for Release-Time Error Input Card [Format (2E12.5)]

Column	Quantity	Description
1-12	DELTR	Mean value of the release-time error
13-24	DELTRS	Mean square value of the release-time error

Table XXVIII. Format for Bomb Component Input Cards [Format (2E11.4)]

Column	Quantity	Description
1-11	FTF(1)	$\dot{x}_e(t_f)$
12-22	FTF(2)	$\dot{h}_e(t_f)$
23-33	FTF(3)	$\dot{u}\ (t_f)$
34-44	FTF(4)	$\dot{\theta}\ (t_f)$
45-55	FTF(5)	$\dot{q}\ (t_f)$
Column	**Quantity**	**Description**
1-11	FTF(6)	$\dot{w}\ (t_f)$
12-22	FTF(7)	$\dot{y}_e(t_f)$
23-33	FTF(8)	$\dot{\psi}\ (t_f)$
34-44	FTF(9)	$\dot{r}\ (t_f)$
45-55	FTF(10)	$\dot{v}\ (t_f)$

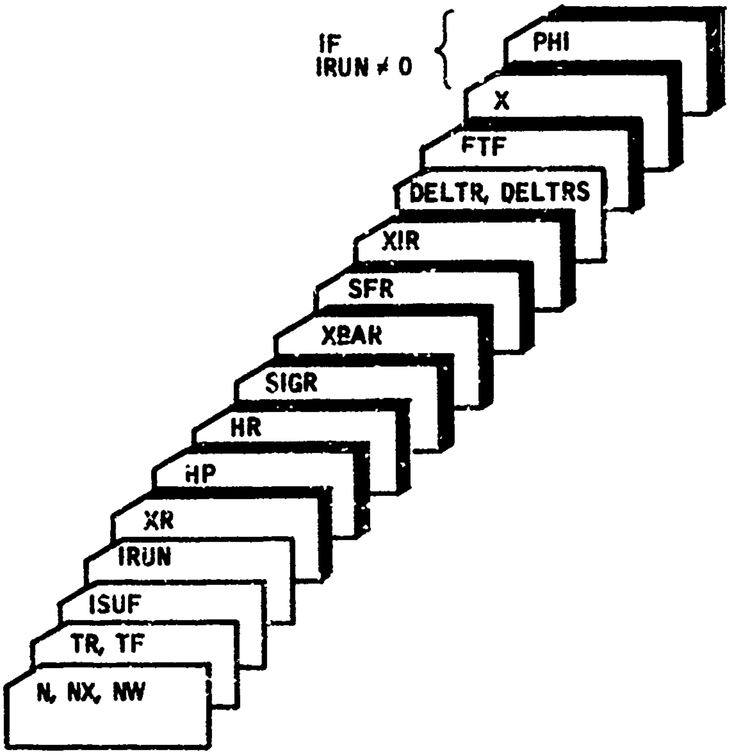


Figure 92. ADAP 3 Input Card Deck

Permanent Disc File Input

Permanent-file input occurs first in the main program, just before the outer loop is entered, and is called for only if IRUN = 0. If it is so, the linear data (F,G) obtained during ADAP 1 run for weapon are read in for the first two time points after release. Data inputs for the subsequent time points occur just before the new outer-loop computations begin.

OUTPUT DESCRIPTION

The output from ADAP 3 is in print and punched-card form.

The input parameters, NX, NW, N, TR, TF and input matrix X_r are printed after card inputs.

The computed quantities, X_f, ϕ, E, V, $\tilde{X}_H$, $\tilde{X}_V$, CEP_H, CEP_V, Q_H, Q_V, $\widetilde{CEP}_H$, $\widetilde{CEP}_V$ are all printed out at the nominal impact time t_f.

The punched-card output occurs at the end of an ADAP 3 run. If IRUN=0, the X_f, ϕ and Q_H matrices are punched. If IRUN $\neq$ 0 only the Q_H matrix is punched.

ADAP 3 PROGRAM DESCRIPTION

ADAP 3 MAIN PROGRAM

The ADAP 3 main program accepts the release covariance of a weapon and propagates it to the impact. Its functional flow diagram is shown in Figure 93. At the start, initial parameters are read and printed out. Then all matrix locations are cleared and the necessary input data for the bomb release covariance computation are input. The release covariance of aircraft X_r is printed out, and the rows and columns of two states corresponding to roll (p,ϕ) are deleted, since the weapon itself does not have roll data and corresponding differential equations. Then the initial mean state and covariance of the bomb are calculated. If IRUN = 0, the covariance differential equation with zero initial condition is integrated to obtain the forced covariance matrix X_f. At the same time the fundamental matrix (transition matrix) of the weapon is computed. This is used to propagate the homogeneous covariance matrix to impact. Both matrices, X_f and ϕ, are printed out at the impact point. Also, the total covariance is computed and printed out.

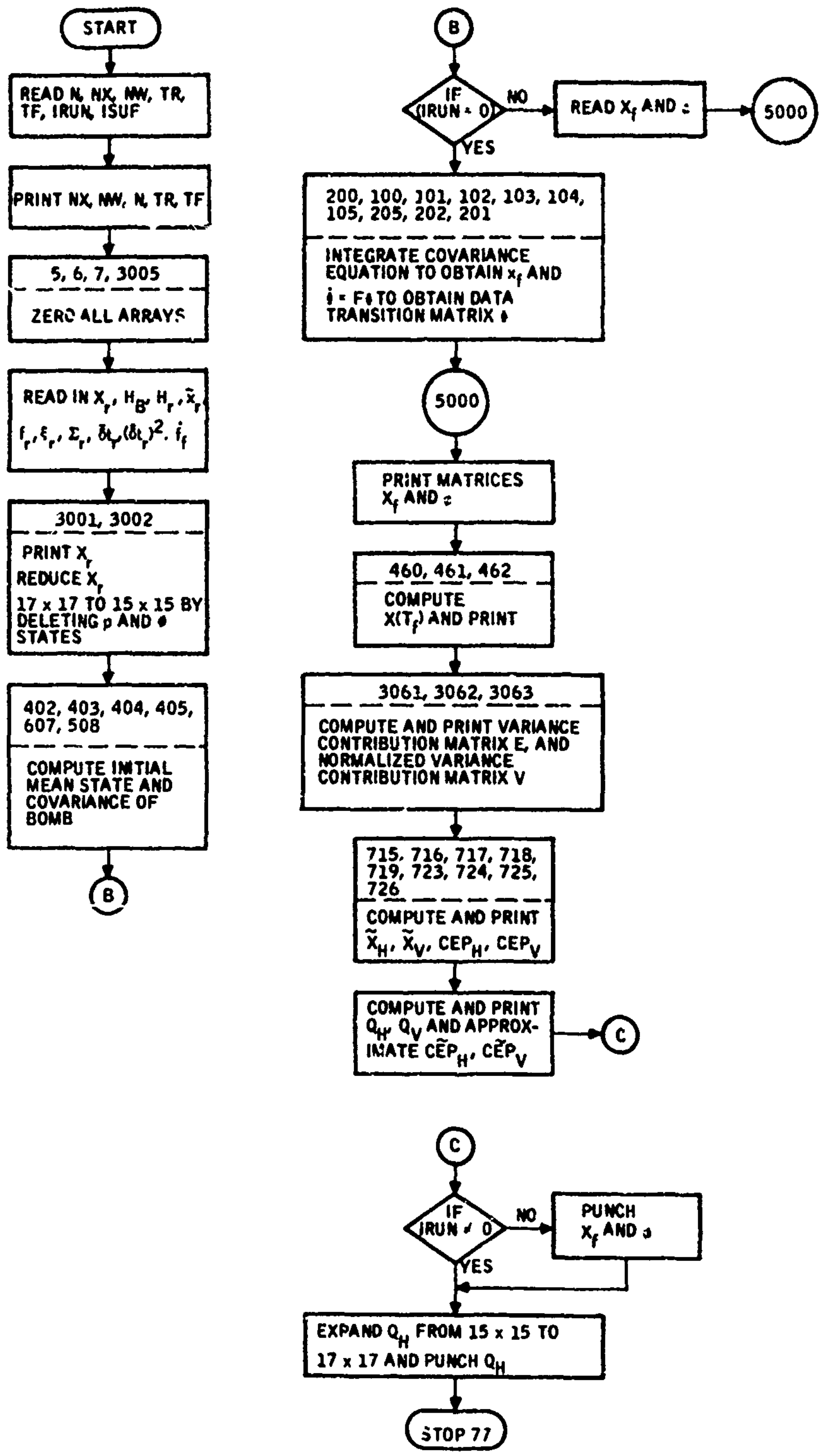


Figure 93. ADAP 3 (PERK) Functional Diagram

If IREAD $\neq 0$ then the program reads X_f and ϕ which are punched on cards from the previous ADAP 3 run, in this case integration is not needed. From the nominal impact covariance several auxiliary data are derived. First the variance contribution and the normalized variance contribution matrices are computed and printed out. Then the horizontal and vertical impact covariance matrices and corresponding CEPs are calculated and printed out. Next the optimal control weighting matrices and approximate CEP calculations are made. Near the end, the IRUN switch is tested. If IRUN = 0, X_f and ϕ are punched. If not only the weighting matrix Q_H is punched on cards after it is expanded to the full matrix corresponding to the aircraft data (rows and columns corresponding to p, ϕ are provided).

The program has two basic loops, the integration loop and the data update loop. Data update time ΔT_u is 1 second. Integration step size Δt is 0.01 second. Data outputting interval ΔT_o is 0.1 second. The time-varying coefficient matrices for integration are obtained by a linear interpolation between data points and are assumed to be constant during the integration interval.

Figure 94 illustrates data updating, integration and outputting processes for arbitrary step sizes.

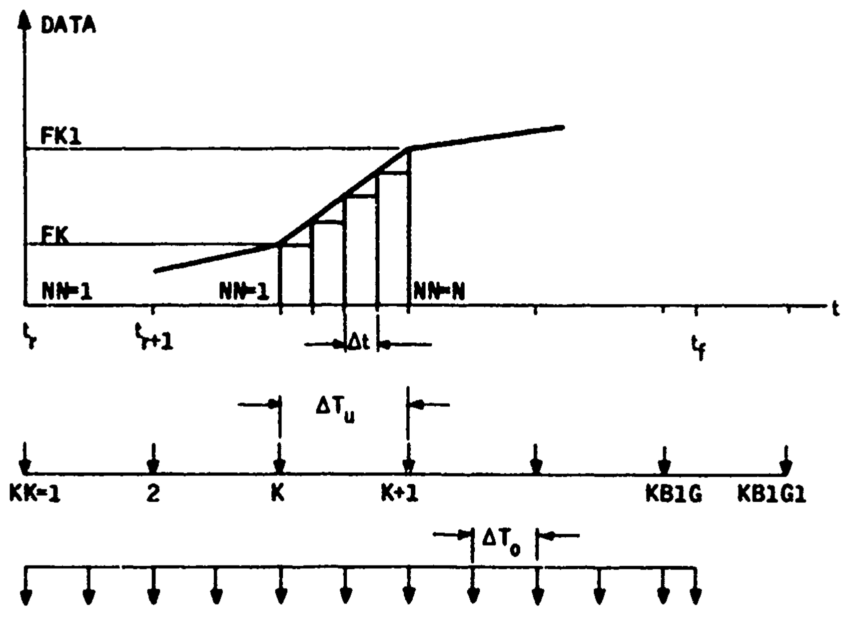


Figure 94. ADAP 3 Data Update, Integration and Outputting

The detailed flow chart for ADAP 3 is shown in Figure 95 and the program listing in Figure 96. Symbols are listed in Table XXIX. ADAP 3 subroutines are listed and described briefly in Table XXX. The auxiliary subroutines INPT, MP and OUTP are similar to those of ADAP 2 described in Section IV.

ADAP 3 SUBROUTINES

Subroutine SHUF

Subroutine SHUF generates the shuffled linear data for the weapon. Its program listing is given in Figure 97 and its symbols are listed in Table XXXI.

Subroutine INTEG

Subroutine INTEG integrates $\dot{X}$ and $\dot{\phi}$ equations for one integration step using Adams open quadrature formula given in Section III of Volume I. Its flow chart is shown in Figure 98 and its program listing in Figure 99. Symbols are listed in Table XXXII.

Subroutine DIFF

Subroutine DIFF generates the forward difference of linear data corresponding to 1-second intervals. It also generates the current value of data at each integration step. Its flow chart is shown in Figure 100 and its program listing in Figure 101. Symbols are listed in Table XXXIII.

Subroutine CEPC

Subroutine CEPC generates the CEP performance measure (circular error probable). It implements the analysis given in Section VIII of Volume I. Its flow diagram is shown in Figure 102 and its program listing in Figure 103. Symbols are listed in Table XXXIV.

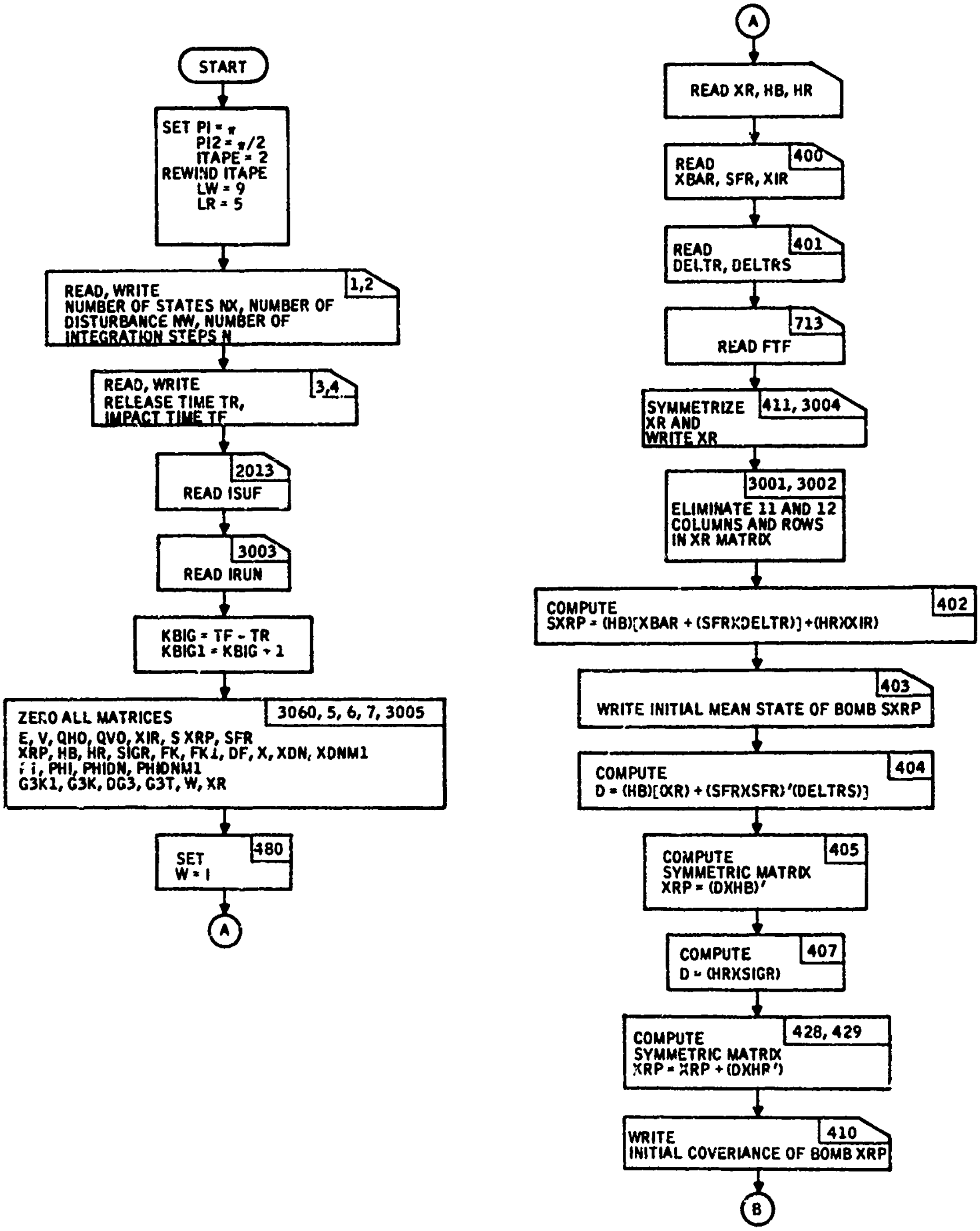


Figure 95. ADAP 3 Main Program Flow Diagram

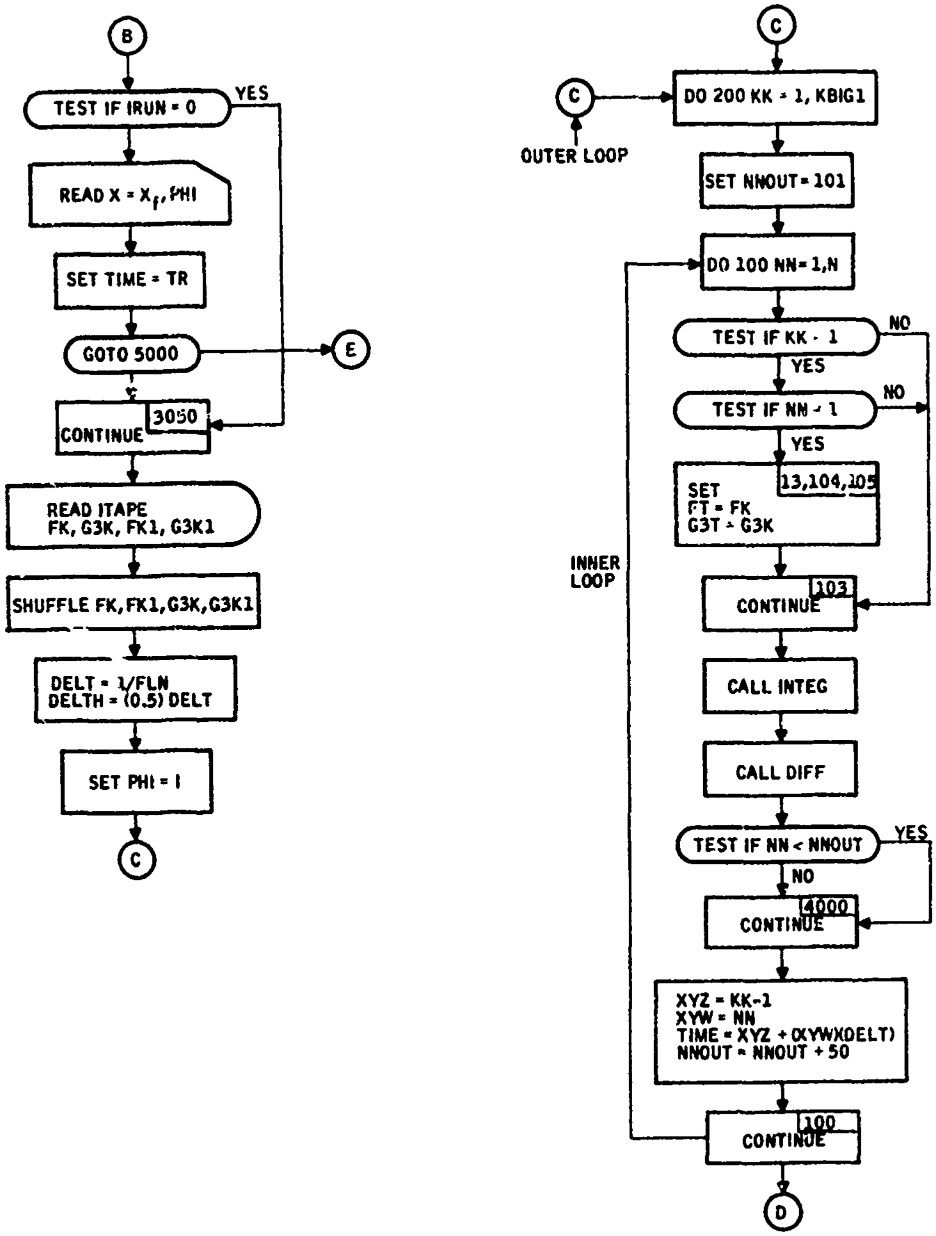


Figure 95. ADAP 3 Main Program Flow Diagram (continued)

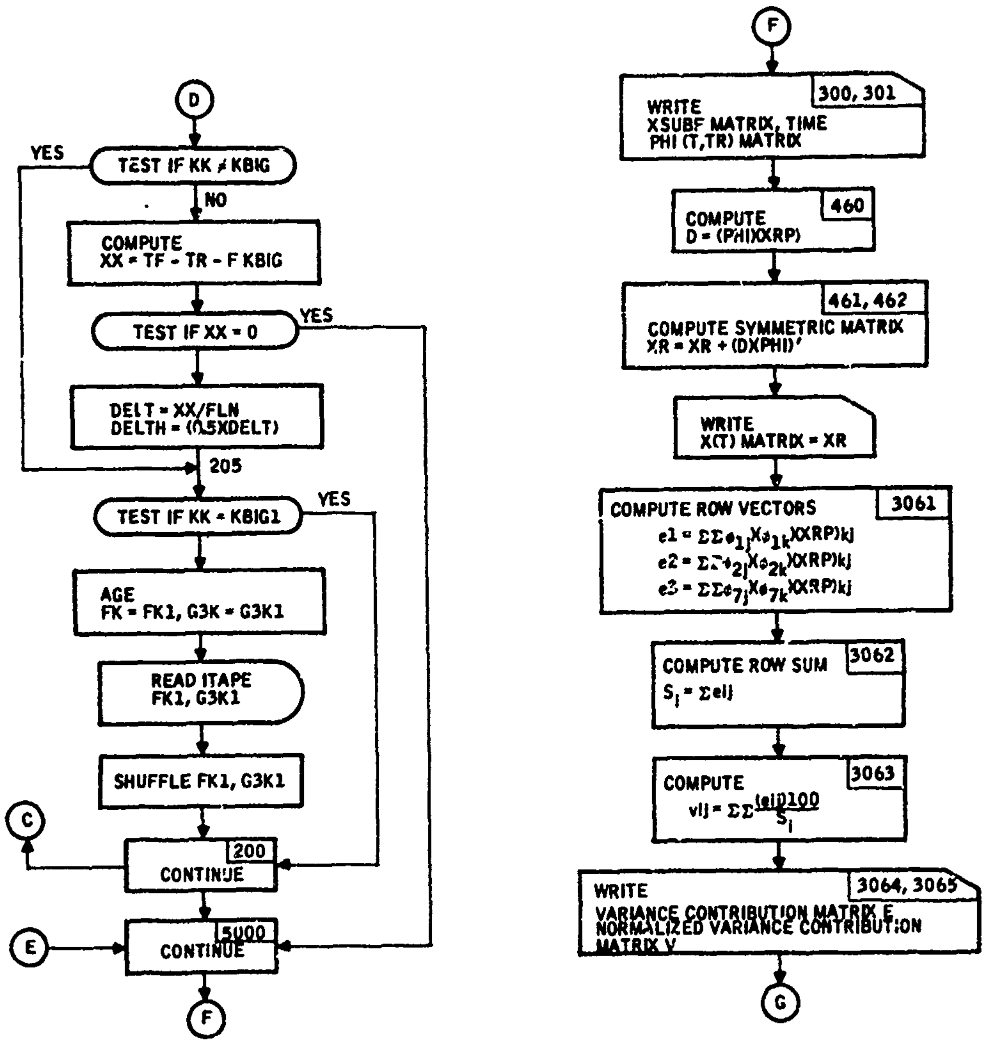


Figure 95. ADAP 3 Main Program Flow Diagram (continued)

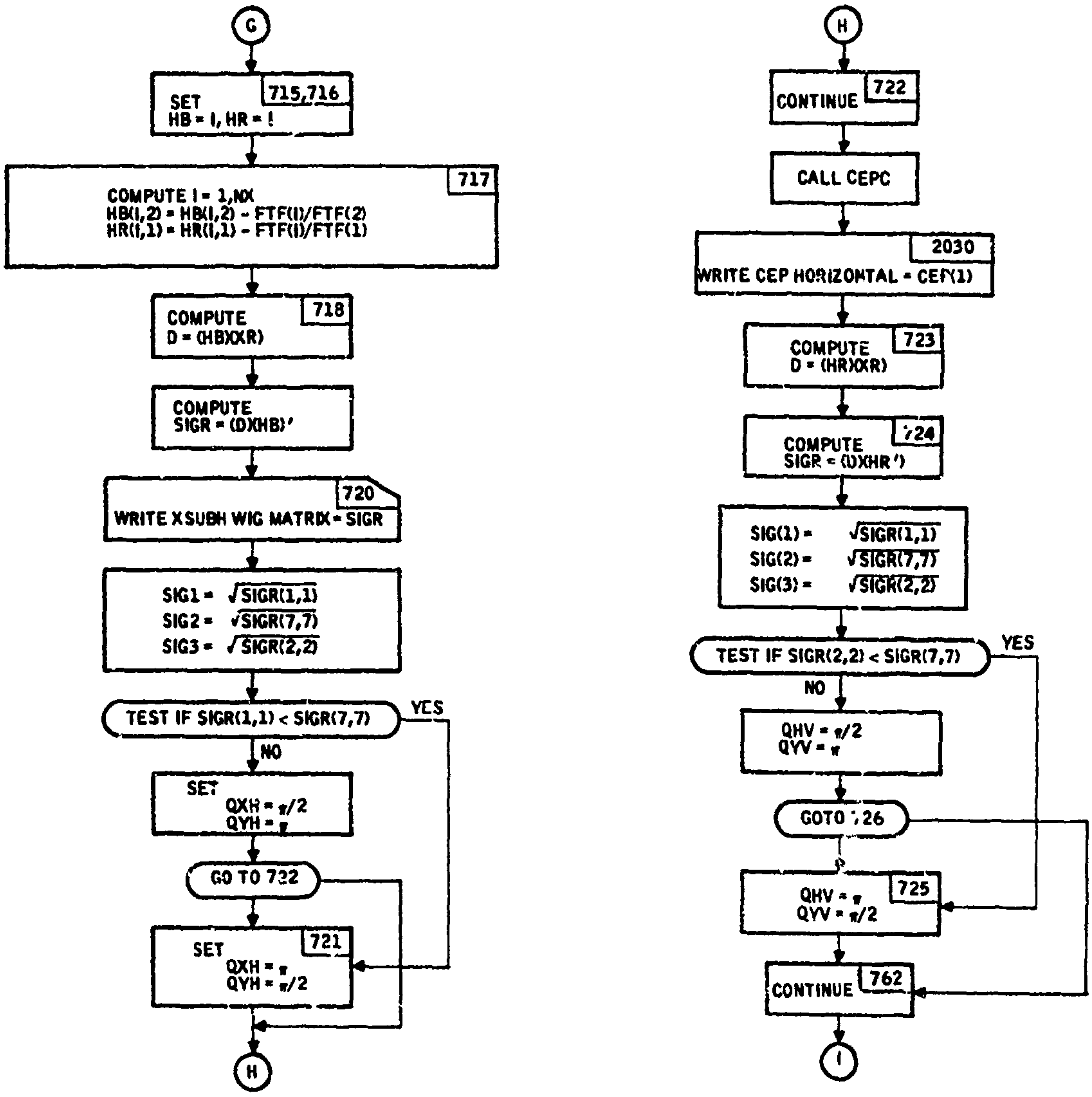


Figure 95. ADAP 3 Main Program Flow Diagram (continued)

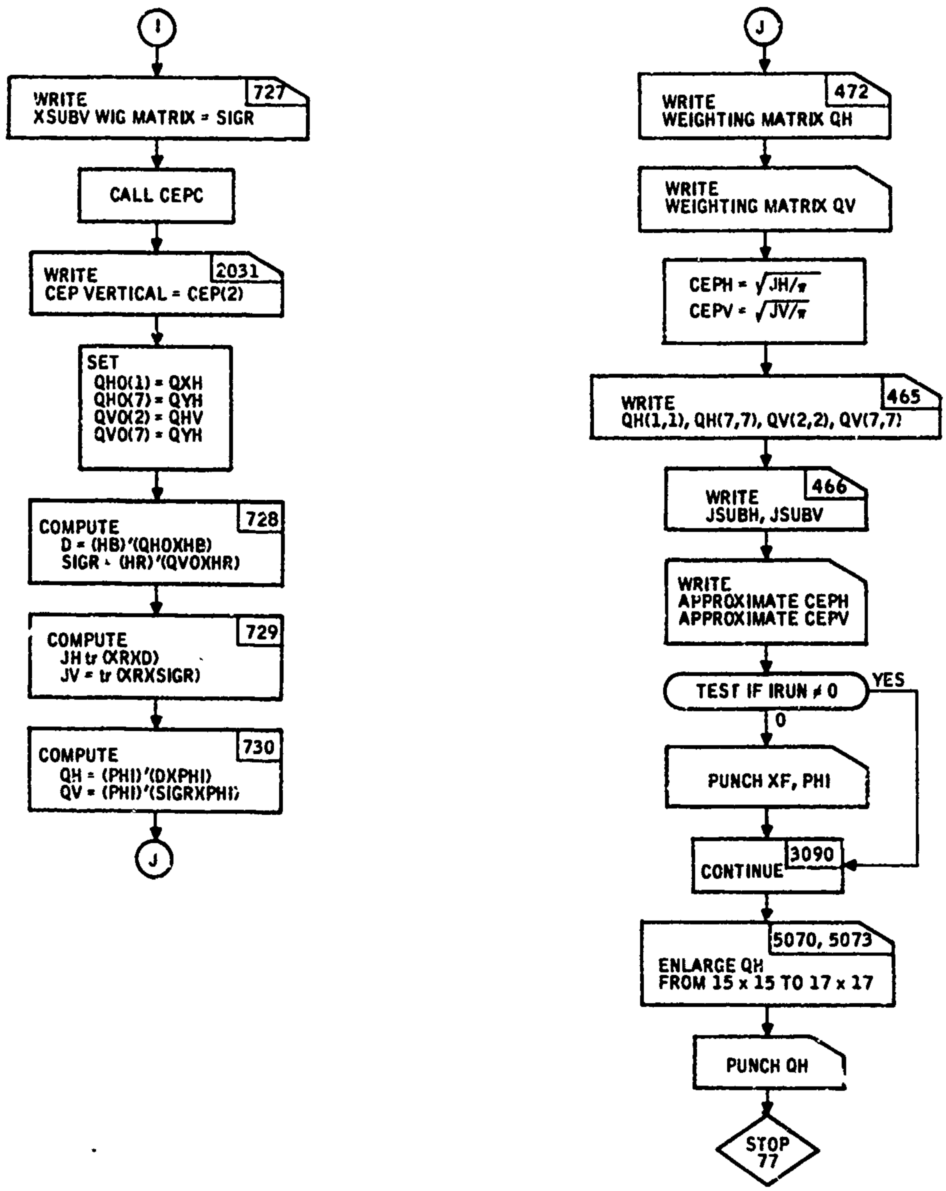


Figure 95. ADAP 3 Main Program Flow Diagram (concluded)

```
      PROGRAM ADAP3(INPUT,OUTPUT,PUNCH,TAPE5=INPUT,TAPE9=OUTPUT,TAPE3=PU
     1NCH,TAPE2)
      DIMENSION E(3,15),V(3,15),S(3)
      DIMENSION D(20,20),ISUF(20)
      DIMENSION FTF(10)
      COMMON FK(20,20),FK1(20,20),FT(15,15),DF(15,15),X(15,15),XR(17,17)
      COMMON XDN(15,15),XDNM1(15,15),PHI(15,15),PHIDN(15,15)
      COMMON PHIDNM1(15,15),G3K(20,4),G3K1(20,4),G3T(20,4),DG3(20,4)
      COMMON C(20,4),W(4,4),XBAR(15),YWB(12),SIG(3),CEP(2)
      COMMON NX,NW,N,TF,TR,KBIG,DELT,DELTH,LW,LR,ITAPE
      COMMON XRP(20,20),SXRP(20),SFR(20),HB(15,15),HR(15,15),SIGR(15,15)
      COMMON XIR(20),QHO(20),QVO(20),QH(20,20),QV(20,20)
      REAL JH,JV
      PI=3.14159265
      PI2=PI/2.
      ITAPE=2
      REWIND ITAPE
      LW=9
      LP=3
      LR=5
      READ(LR,1) N,NX,NW
    1 FORMAT(3I3)
      WRITE(LW,2)NX,NW,N
    2 FORMAT(1H1/7X,18H NUMBER OF STATES=I2/7X,24H NUMBER OF DISTURBANCE
     1S=I2/7X,34H NUMBER OF INTEGRATION STEPS/SEC.=I3/)
      READ(LR,3)TR,TF
    3 FORMAT(2E15.8)
      WRITE(LW,4)TR,TF
    4 FORMAT(/7X,14H RELEASE TIME=E15.8/7X,13H IMPACT TIME=E15.8/)
      READ(5,2013)ISUF
 2013 FORMAT(20I2)
      READ(LR,3003)IRUN
 3003 FORMAT(I2)
      KBI  TF-TR
      KBIG =KBIG+1
C ZERO ALL ARRAYS
C
      DO 3060 I=1,3
      S(I)=0.
      DO 3060 J=1,15
      E(I,J)=0.
      V(I,J)=0.
 3060 CONTINUE
      DO 6 I=1,NX
      QHO(I)=0.
      QVO(I)=0.
      XIR(I)=0.
      SXRP(I)=0.
      SFR(I)=0.
      DO 5 J=1,NX
      XRP(I,J)=0.
      HB(I,J)=0.
      HR(I,J)=0.
      SIGR(I,J)=0.
```

Figure 96. ADAP 3 Main Program Input/Output Listing

```
      FK(I,J)=0.
      FK1(I,J)=0.
      DF(I,J)=0.
      X(I,J)=0.
      XDN(I,J)=0.
      XDNM1(I,J)=0.
      FT(I,J)=0.
      PHI(I,J)=0.
      PHIDN(I,J)=0.
    5 PHIDNM1(I,J)=0.
      DO 6 J=1,NW
      G3K1(I,J)=0.
      G3K(I,J)=0.
      DG3(I,J)=0.
    6 G3T(I,J)=0.
      DO 7 I=1,NW
      DO 7 J=1,NW
    7 W(I,J)=0.
      DO 480 I=1,NW
  480 W(I,I)=1.
      DO 3005 I=1,17
      DO 3005 J=1,17
 3005 XR(I,J)=0.
C
C READ IN X ,H ,H ,X ,F XI ,SIG ,DELT
C          R  B  R  R  R  R    R     R
      CALL INPT(XR,17,17)
      CALL INPT(HB,15,15)
      CALL INPT(HR,15,15)
      CALL INPT(SIGR,15,15)
      READ(LR,400)(XBAR(I),I=1,NX)
      READ(LR,400)(SFR(I),I=1,NX)
      READ(LR,400)(XIR(I),I=1,NX)
  400 FORMAT(5E12.5)
      READ(LR,401)DELTR,DELTRS
  401 FORMAT(2E12.5)
      READ(LR,713)(FTF(I),I=1,10)
  713 FORMAT(5E11.4)
      DO 411 I=1,17
      DO 411 J=I,17
  411 XR(J,I)=XR(I,J)
      WRITE(LW,3004)
 3004 FORMAT(1H1/7X,10H XR MATRIX/)
      CALL MP(17,17,NX,NX,XR)
C REDUCE XR FROM 17BY17 TO 15BY15
      DO 3001 I=1,10
      DO 3001 J=13,17
      JJ=J-2
      XR(I,JJ)=XR(I,J)
      XR(JJ,I)=XR(I,J)
 3001 CONTINUE
      DO 3002 I=13,17
      II=I-2
      DO 3002 J=13,17
```

Figure 96. ADAP 3 Main Program Input/Output Listing (continued)

```
      JJ=J-2
 3002 XR(II,JJ)=X(I,J)
C
C  COMPUTE INITIAL STATE MEAN AND COVARIANCE OF BOMB
C
      DO 402 I=1,NX
      SXRP(I)=0.
      DO 402 J=1,NX
  402 SXRP(I)=SXRP(I)+HB(I,J)*(XBAR(J)+SFR(J)*DELTR)+HR(I,J)*XIR(J)
      WRITE(LW,403)(SXRP(I),I=1,NX)
  403 FORMAT(1H1/7X,27H INITIAL MEAN STATE OF BOMB/(E25.8))
      DO 404 I=1,NX
      DO 404 J=1,NX
      D(I,J)=0.
      DO 404 K=1,NX
  404 D(I,J)=D(I,J)+HB(I,K)*(XR(K,J)+SFR(K)*SFR(J)*DELTRS)
      DO 405 I=1,NX
      DO 405 J=I,NX
      XRP(I,J)=0.
      XRP(J,I)=0.
      DO 405 K=1,NX
  405 XRP(I,J)=XRP(I,J)+D(I,K)*HB(J,K)
      DO 407 I=1,NX
      DO 407 J=1,NX
      D(I,J)=0.
      DO 407 K=1,NX
  407 D(I,J)=D(I,J)+HR(I,K)*SIGR(K,J)
      DO 408 I=1,NX
      DO 408 J=I,NX
      DO 409 K=1,NX
  409 XRP(I,J)=XRP(I,J)+D(I,K)*HR(J,K)
  408 XRP(J,I)=XRP(I,J)
      WRITE(LW,410)
  410 FORMAT(1H1/7X,27H INITIAL COVARIANCE OF BOMB/)
      CALL MP(20,20,NX,NX,XRP)
C
C READ F(TR),F(TR+1),G3(TR),G3(TR+1)
C
      IF(IRUN.EQ.0) GOTO 3050
      CALL INPT(X,15,15)
      CALL INPT(PHI,15,15)
      TIME=TR
      GOTO 5000
 3050 CONTINUE
      READ(ITAPE)FK
      READ(ITAPE)G3K
      READ(ITAPE)FK1
      READ(ITAPE)G3K1
      CALL SHUF(FK,20,20,1,ISUF,20,20,D)
      CALL SHUF(FK1,20,20,1,ISUF,20,20,D)
      CALL SHUF(G3K,20,4,2,ISUF,20,4,D)
      CALL SHUF(G3K1,20,4,2,ISUF,20,4,D)
      FLN=N
      DELT=1./FLN
```

Figure 96. ADAP 3 Main Program Input/Output Listing (continued)

```
      DELTH=.5*DELT
      DO 8 I=1,NX
    8 PHI(I,I)=1.
      DO 200 KK=1,KBIG1
      NNOUT=101
      DO 100 NN=1,N
      IF(KK-1)103,101,103
  101 IF(NN-1)103,102,103
  102 DO 104 I=1,NX
      DO 105 J=1,NX
  105 FT(I,J)=FK(I,J)
      DO 104 J=1,NW
  104 G3T(I,J)=G3K(I,J)
  103 CONTINUE
      CALL INTEG(KK,NN)
      CALL DIFF(KK,NN)
      IF(NN.LT.NNOUT) GOTO 4000
 4000 CONTINUE
      XYZ=KK-1
      XYW=NN
      TIME=XYZ+XYW*DELT
      NNOUT=NNOUT+50
  100 CONTINUE
      IF(KK.NE.KBIG) GOTO 205
      FKBIG=KBIG
      XX=TF-TR-FKBIG
      IF(XX.EQ.0.) GOTO 5000
      DELT=XX/FLN
      DELTH=.5*DELT
  205 IF(KK.EQ.KBIG1) GOTO 200
C
C AGE DATA POINT,AND READ IN NEXT DATA
C
      DO 202 I=1,NX
      DO 201 J=1,NX
  201 FK(I,J)=FK1(I,J)
      DO 202 J=1,NW
  202 G3K(I,J)=G3K1(I,J)
      READ(ITAPE)FK1
      READ(ITAPE)G3K1
      CALL SHUF(FK1,20,20,1,ISUF,20,20,D)
      CALL SHUF(G3K1,20,4,2,ISUF,20,4,D)
  200 CONTINUE
 5000 CONTINUE
      WRITE(LW,300)TIME
  300 FORMAT(1H1/7X,16H XSUBF MATRIX T=F8.2/)
      CALL MP(15,15,NX,NX,X)
      WRITE(LW,301)
  301 FORMAT(1H1/7X,18H PHI(T ,TR) MATRIX/)
      CALL MP(15,15,NX,NX,PHI)
C
C COMPUTE X(T) TOTAL
C
      DO 460 I=1,NX
```

Figure 96. ADAP 3 Main Program Input/Output Listing (continued)

```
      DO 460 J=1,NX
      D(I,J)=0.
      DO 460 K=1,NX
  460 D(I,J)=D(I,J)+PHI(I,K)*XRP(K,J)
      DO 461 I=1,NX
      DO 461 J=I,NX
      XR(I,J)=X(I,J)
      DO 462 K=1,NX
  462 XR(I,J)=XR(I,J)+D(I,K)*PHI(J,K)
  461 XR(J,I)=XR(I,J)
      WRITE(LW,463)
  463 FORMAT(1H1/7X,12H X(T) MATRIX/)
      CALL MP(17,17,NX,NX,XR)
      DO 3061 J=1,15
      DO 3061 LL=1,15
      E(1,J)=E(1,J)+PHI(1,J)*PHI(1,LL)*XRP(LL,J)
      E(2,J)=E(2,J)+PHI(2,J)*PHI(2,LL)*XRP(LL,J)
      E(3,J)=E(3,J)+PHI(7,J)*PHI(7 LL)*XRP(LL,J)
 3061 CONTINUE
      DO 3062 I=1,3
      S(I)=0.
      DO 3062 J=1,15
 3062 S(I)=S(I)+E(I,J)
      DO 3063 I=1,3
      DO 3063 J=1,15
      V(I,J)=0.
 3063 V(I,J)=E(I,J)*100./S(I)
      WRITE(LW,3064)
 3064 FORMAT(1H1/7X,29H VARIANCE CONTRIBUTION MATRIX/)
      CALL MP(3,15,3,15,E)
      WRITE(LW,3065)
 3065 FORMAT(////7X,40H NORMALIZED VARIANCE CONTRIBUTION MATRIX/)
      CALL MP(3,15,3,15,V)
      DO 715 I=1,NX
      DO 716 J=1,NX
      HB(I,J)=0.
  716 HR(I,J)=0.
      HB(I,I)=1.
  715 HR(I,I)=1.
      DO 717 I=1,NX
      HB(I,2)=HB(I,2)-FTF(I)/FTF(2)
  717 HR(I,1)=HR(I,1)-FTF(I)/FTF(1)
      DO 718 I=1,NX
      DO 718 J=1,NX
      D(I,J)=0.
      DO 718 K=1,NX
  718 D(I,J)=D(I,J)+HB(I,K)*XR(K,J)
      DO 719 I=1,NX
      DO 719 J=1,NX
      SIGR(I,J)=0.
      DO 719 K=1,NX
  719 SIGR(I,J)=SIGR(I,J)+D(I,K)*HB(J,K)
      WRITE(LW,720)
  720 FORMAT(1H1/7X,17H XSUBH WIG MATRIX/)
```

Figure 96. ADAP 3 Main Program Input/Output Listing (continued)

```
      CALL MP(15,15,15,15,SIGR)
      SIG(1)=SQRT(SIGR(1,1))
      SIG(2)=SQRT(SIGR(7,7))
      SIG(3)=SQRT(SIGR(2,2))
      IF(SIGR(1,1).LT.SIGR(7,7)) GOTO 721
      QXH=PI2
      QYH=PI
      GOTO 722
  721 QXH=PI
      QYH=PI2
  722 CONTINUE
      CALL CEPC(SIG,CEP)
      WRITE(LW,2030)CEP(1)
 2030 FORMAT(////7X,16H CEP HORIZONTAL=E15.8/)
      DO 723 I=1,NX
      DO 723 J=1,NX
      D(I,J)=0.
      DO 723 K=1,NX
  723 D(I,J)=D(I,J)+HR(I,K)*XR(K,J)
      DO 724 I=1,NX
      DO 724 J=1,NX
      SIGR(I,J)=0.
      DO 724 K=1,NX
  724 SIGR(I,J)=SIGR(I,J)+D(I,K)*HR(J,K)
      SIG(1)=SQRT(SIGR(1,1))
      SIG(2)=SQRT(SIGR(7,7))
      SIG(3)=SQRT(SIGR(2,2))
      IF(SIGR(2,2).LT.SIGR(7,7)) GOTO 725
      QHV=PI2
      QYV=PI
      GOTO 726
  725 QHV=PI
      QYV=PI2
  726 CONTINUE
      WRITE(LW,727)
  727 FORMAT(1H1/7X,17H XSUBV WIG MATRIX/)
      CALL MP(15,15,15,15,SIGR)
      CALL CEPC(SIG,CEP)
      WRITE(LW,2031)CEP(2)
 2031 FORMAT(////7X,14H CEP VERTICAL=E15.8/)
      QHO(1)=QXH
      QHO(7)=QYH
      QVO(2)=QHV
      QVO(7)=QYV
      DO 728 I=1,NX
      DO 728 J=1,NX
      D(I,J)=0.
      SIGR(I,J)=0.
      DO 728 K=1,NX
      D(I,J)=D(I,J)+HB(K,I)*HR(K,J)*QHO(K)
  728 SIGR(I,J)=SIGR(I,J)+HR(K,I)*HR(K,J)*QVO(K)
      JH=0.
      JV=0.
      DO 729 I=1,NX
```

Figure 96. ADAP 3 Main Program Input/Output Listing (continued)

```
      DO 729 J=1,NX
      JH=JH+XR(I,J)*D(J,I)
  729 JV=JV+XR(I,J)*SIGR(J,I)
      DO 730 I=1,NX
      DO 730 J=1,NX
      QH(I,J)=0.
      QV(I,J)=0.
      DO 730 M=1,NX
      DO 730 K=1,NX
      QH(I,J)=QH(I,J)+PHI(M,I)*D(M,K)*PHI(K,J)
  730 QV(I,J)=QV(I,J)+PHI(M,I)*SIGR(M,K)*PHI(K,J)
      WRITE(LW,472)
  472 FORMAT(1H1/7X,20H WEIGHTING MATRIX QH/)
      CALL MP(20,20,NX,NX,QH)
      WRITE(LW,473)
  473 FORMAT(1H1/7X,20H WEIGHYING MATRIX QV/)
      CALL MP(20,20,NX,NX,QV)
      CEPH=SQRT(JH/PI)
      CEPV=SQRT(JV/PI)
      WRITE(LW,465)QH0(1),QH0(7),QV0(2),QV0(7)
  465 FORMAT(1H1/7X,10H QH(1,1) =E15.8,10H QH(7,7) =E15.8,10H QV(2,2) =E
     115.8,10H QV(7,7) =E15.8/)
      WRITE(LW,466)JH,JV
  466 FORMAT(///7X,7H JSUBH=E15.8,7H JSUBV=E15.8/)
      WRITE(LW,467)CEPH,CEPV
  467 FORMAT(//7X,18H APPROXIMATE CEPH=E15.8,18H APPROXIMATE CEPV=E15.8/
     1)
      IF(IRUN.NE.0) GOTO 3090
C PUNCH XF
      CALL OUTP(15,15,NX,NX,X,LP)
C PUNCH PHI
      CALL OUTP(15,15,NX,NX,PHI,LP)
 3090 CONTINUE
C
C ENLARGE QH FROM 15X15 TO 17X17 BEFORE PUNCHING
C
      DO 5070 I=1,15
      DO 5070 J=1,15
 5070 D(I,J)=QH(I,J)
      DO 5071 I=1,20
      DO 5071 J=1,20
 5071 QH(I,J)=0.
      DO 5081 I=1,10
      DO 5081 J=1,10
 5081 QH(I,J)=D(I,J)
      DO 5072 I=13,17
      II=I-2
      DO 5072 J=13,17
      JJ=J-2
 5072 QH(I,J)=D(II,JJ)
      DO 5073 I=1,10
      DO 5073 J=13,17
      JJ=J-2
      QH(I,J)=D(I,JJ)
 5073 QH(J,I)=D(I,JJ)
      CALL OUTP(20,20,17,17,QH,LP)
      STOP 77
      END
```

Figure 96. ADAP 3 Main Program Input/Output Listing (concluded)

Table XXIX. List of Symbols for ADAP 3 (PERK)

Quantity	Mnemonic	Initial Value	Input	Description
	C(I, J)			Working matrix
	CEP(I)			Vector containing horizontal and vertical CEP
	D(I, J)			Working matrix
Δt	DELT			Integration step size: $\Delta t = 1./FLN$
$\frac{\Delta t}{2}$	DELTH			1/2 integration step size
$(F(t_{K+1}) - F(t_K))\Delta t$	DF(I, J)			Forward difference of matrix F
$(G_3(t_{K+1}) - G_3(t_K))\Delta t$	DG3(I, J)			Forward difference of matrix G_3
$F(t_K)$	FK(I, J)		X	Matrix F at t_K
$F(t_{K+1})$	FK1(I, J)		X	Matrix F at t_{K+1}
	FKBIL			KBIG floated
	FLN			N floated
$F(t)$	FT(I, J)			Matrix F at each integration step
$G_3(t_K)$	G3K(I, J)		X	Matrix G_3 at t_K
$G_3(t_{K+1})$	G3K1(I, J)		X	Matrix G_3 at t_{K+1}
$G_3(t)$	G3T(I, J)			Matrix G_3 at each integration step
	ICEP		X	Switch: $= 0 \Rightarrow$ compute weight; $\neq 0 \Rightarrow$ compute CEP
	ISUF(I)		X	Integer vector used to shuffle matrices

Table XXIX. List of Symbols for ADAP 3 (PERK) (continued)

Quantity	Mnemonic	Initial Value	Input	Description
	ITAPE	2		Logical number of data tape
	KBIG			KBIG integer $[t_f - t_R]$
	KBIG1			KBIG1 = KBIG + 1
	LR	5		Logical number input tape: Set
	LW	9	X	Logical number output tape: Set
	N			Number of times through inner integration loop
	NNOUT	10		Counter for output in inner integration loop: Set
	NW		X	Number of disturbances
	NX		X	Order covariance equation
ϕ	PHI(I, J)		X	State transition matrix
$\dot{\phi}_n$	PHIDN(I, J)			Current derivative of state transition matrix
$\dot{\phi}_{n-1}$	PHIDNM1 (I, J)			Past derivative of state transition matrix
σ	SIG(I)			Vector containing the square root of the variances of x, y, and z at impact
t_f	TF		X	Impact time
t	TIME			Running time
t_r	TR		X	Release time
	W(I, J)			
X	X(I, J)			State covariance

Table XXIX. List of Symbols for ADAP 3 (PERK) (concluded)

Quanity	Mnemonic	Initial Value	Input	Description
$\bar{x}$	XBAR(I)			Mean output
$\dot{X}_n$	XDN(I, J)			Current derivative of state covariance
$\dot{X}_{n-1}$	XDNM1ΔI, J)			Past derivative of state covariance
X_r	XR(I, J)		X	Covariance at release
$\bar{y}_w$	YWB(I)		X	Mean input
	NN			Integer index NN = 1, 2, . . . , N inner loop
	KK			Integer index KK = 1, 2, . . . , KBIG1 outer loop

Table XXX. ADAP 3 Subroutine Summary

Subroutine	Description	Flow Diagram (Figure No.)	Program Listing (Figure No.)	List of Symbols (Table No.)	Volume I Reference
ADAP 3 'PERK) Main Program)	Main program for nonstationary weapon performance evaluation	95	96	XXIX	pp. 125-148
SHUF	Data shuffler	--	97	XXXI	--
INTEG	Integrator for fundamental matrix and covariance	98	99	XXXII	p. 40
DIFF	Current value of data	100	101	XXXIII	--
CEPC	CEP calculation	102	103	XXXIV	p. 140
INPT	Matrix input	--	--	--	--
MP	Matrix print	--	--	--	--
OUTP	Matrix punch	--	--	--	--

```
      SUBROUTINE SHUF(A,NR,NC,IRC,ISUF,NX,NY,D)
      DIMENSION A(NR,NC),D(20,20),ISUF(20)
      GOTO(1,10,20),IRC
    1 CONTINUE
      DO 2 I=1,NX
      II=ISUF(I)
      DO 2 J=1,NY
    2 D(I,J)=A(II,J)
      DO 3 I=1,NX
      DO 3 J=1,NY
      JJ=ISUF(J)
    3 A(I,J)=D(I,JJ)
      GOTO 20
   10 CONTINUE
      DO 40 I=1,NX
      D(I,1)=A(I,2)
      D(I,2)=A(I,3)
      A(I,2)=D(I,2)
   40 A(I,3)=D(I,1)
      DO 50 I=1,NX
      II=ISUF(I)
      DO 50 J=1,4
   50 D(I,J)=A(II,J)
      DO 55 I=1,NX
      DO 55 J=1,4
   55 A(I,J)=D(I,J)
   20 CONTINUE
      RETURN
      END
```

Figure 97. Subroutine SHUF Program Listing

Table XXXI. List of Symbols for Subroutine SHUF

Mnemonic	Input	Description
A(I, J)		Matrix to be shuffled
D(I, J)		Working matrix
ISUF		Vector containing indices in the desired order
NC		Number of columns in A
NR		Number of rows in A
NX		Number of rows to be shuffled
NY		Number of columns to be shuffled

Table XXXII. List of Symbols for Subroutine INTEG

Mnemonic	Input	Description
C1		Constant used in integration formula when: KK ≡NN ≡ 1 C1 = 2 Otherwise C1 = 3
KK		Integer index KK = 1, 2, ..., KBIG1
NN		Integer index NN = 1, 2, ..., N; all other symbols common with main program PERK

Table XXXIII. List of Symbols for Subroutine DIFF

Mnemonic	Input	Description
NN		Integer index, i.e., NN = 1, 2, ..., N when NN ≡ 1 ⇒ compute new DF and DG3 matrices and then update F and G 3: NN ≠1 ⇒ update F and G3 . . . . All other symbols common with main program PERK

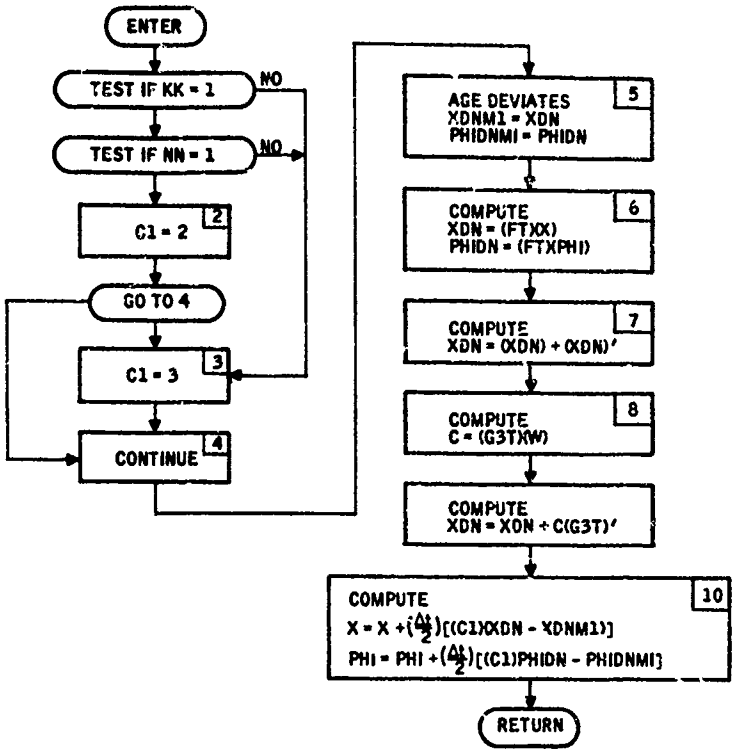


Figure 98. Subroutine INTEG Flow Diagram

```
      SUBROUTINE INTEG(KK,NN)
      COMMON FK(20,20),FX1(20,20),FT(15,15),DF(15,15),X(15,15),XR(17,17)
      COMMON XDN(15,15),XDNM1(15,15),PHI(15,15),PHIDN(15,15)
      COMMON PHIDNM1(15,15),G3K(20,4),G3K1(20,4),G3T(20,4),DG3(20,4)
      COMMON C(20,4),W(4,4),XBAR(15),YWB(12),SIG(3),CEP(2)
      COMMON NX,NW,N,TF,TR,KBIG,DELT,DELTH,LW,LR,ITAPE
      COMMON XRP(20,20),SXRP(20),SFR(20),HB(15,15),HR(15,15),SIGR(15,15)
      COMMON XIR(20),QHO(20),QVO(20),QH(20,20),QV(20,20)
      IF(KK-1)3,1,3
    1 IF(NN-1)3,2,3
    2 C1=2.
      GOTO 4
    3 C1=3.
    4 CONTINUE
C
C AGE DERIVATIVES
C
      DO 5 I=1,NX
      DO 5 J=1,NX
      XDNM1(I,J)=XDN(I,J)
    5 PHIDNM1(I,J)=PHIDN(I,J)
C
C COMPUTE DERIVATIVES
C
      DO 6  I=1,NX
      DO 6  J=1,NX
      XDN(I,J)=0.
      PHIDN(I,J)=0.
      DO 6  K=1,NX
      XDN(I,J)=XDN(I,J)+FT(I,K)*X(K,J)
    6 PHIDN(I,J)=PHIDN(I,J)+FT(I,K)*PHI(K,J)
C
      DO 7 I=1,NX
      DO 7 J=I,NX
      XDN(I,J)=XDN(I,J)+XDN(J,I)
    7 XDN(J,I)=XDN(I,J)
C
      DO 8 I=1,NX
      DO 8 J=1,NW
      C(I,J)=0.
      DO 8 K=1,NW
    8 C(I,J)=C(I,J)+G3T(I,K)*W(K,J)
C
      DO 9 I=1,NX
      DO 9 J=1,NX
      DO 9 K=1,NW
    9 XDN(I,J)=XDN(I,J)+C(I,K)*G3T(J,K)
C
C INTEGRATE
C
      DO 10 I=1,NX
      DO 10 J=1,NX
      X(I,J)=X(I,J)+DELTH*(C1*XDN(I,J)-XDNM1(I,J))
   10 PHI(I,J)=PHI(I,J)+DELTH*(C1*PHIDN(I,J)-PHIDNM1(I,J))

   RETURN
   END
```

Figure 99. Subroutine INTEG Program Listing

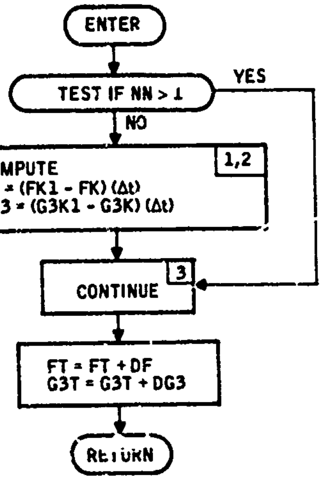


Figure 100. Subroutine DIFF Flow Diagram

```
      SUBROUTINE DIFF(KK,NN)
      COMMON FK(20,20),FK1(20,20),FT(15,15),DF(15,15),X(15,15),XK(17,17)
      COMMON XDN(15,15),XDNM1(15,15),PHI(15,15),PHIDN(15,15)
      COMMON PHIDNM1(15,15),G3K(20,4),G3K1(20,4),G3T(20,4),DG3(20,4)
      COMMON C(20,4),W(4,4),XBAR(15),YWB(12),SIG(3),CEP(2)
      COMMON NX,NW,N,TF,TR,KBIG,DELT,DELTH,LW,LR,ITAPE
      COMMON XRP(20,20),SXRP(20),SFR(20),HB(15,15),HR(15,15),SIGR(15,15)
      COMMON XIR(20),QHO(20),QVO(20),QH(20,20),QV(20,20)
      IF(NN.GT.1) GOTO 3
      DO 1 I=1,NX
      DO 2 J=1,NX
      DF(I,J)=0.
    2 DF(I,J)=(FK1(I,J)-FK(I,J))*DELT
      DO 1 J=1,NW
      DG3(I,J)=0.
    1 DG3(I,J)=(G3K1(I,J)-G3K(I,J))*DELT
    3 CONTINUE
      DO 4 I=1,NX
      DO 5 J=1,NX
    5 FT(I,J)=FT(I,J)+DF(I,J)
      DO 4 J=1,NW
    4 G3T(I,J)=G3T(I,J)+DG3(I,J)
      RETURN
      END
```

Figure 101. Subroutine DIFF Program Listing

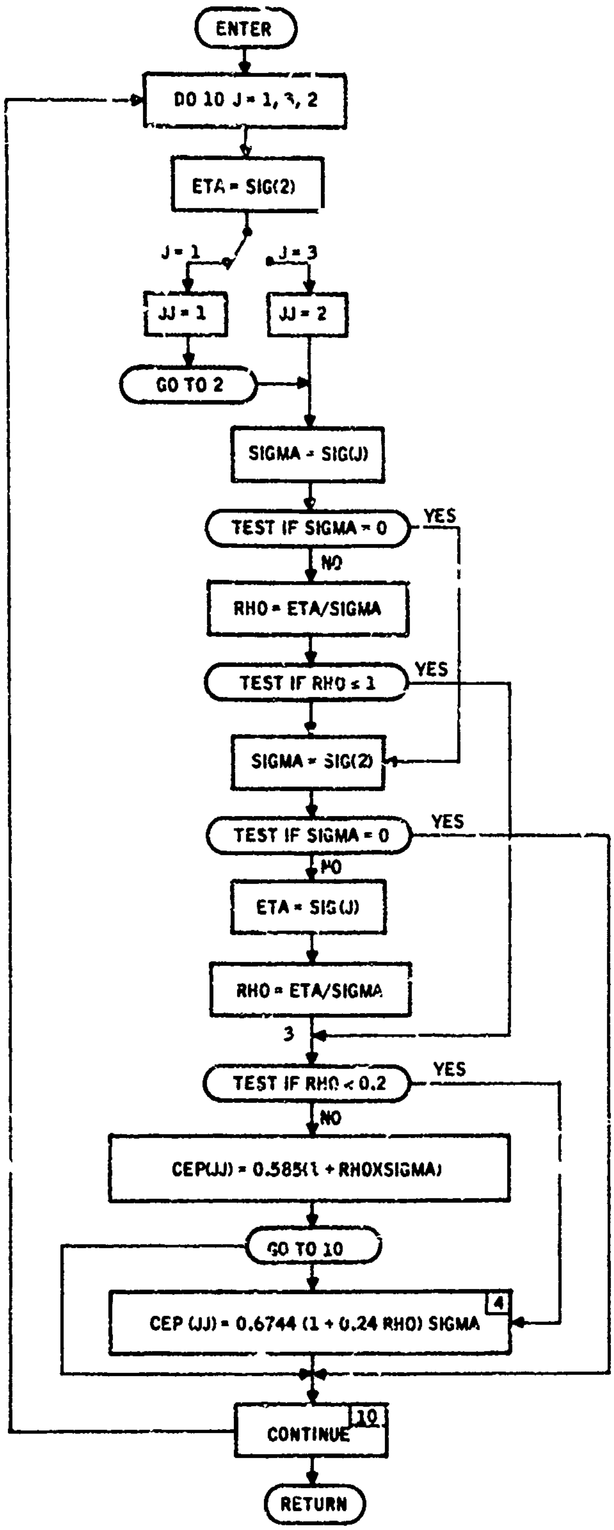


Figure 102. Subroutine CEPC Flow Diagram

```
      SUBROUTINE CEPC(SIG,CEP)
      DIMENSION SIG(3),CEP(2)
      DO 10 J=1,3,2
      ETA=SIG(2)
      IF(J.GT.1) GOTO 1
      JJ=J
      GOTO 2
1     JJ=2
2     CONTINUE
      SIGMA=SIG(J)
      IF(SIGMA.EQ.0.) GOTO 5
      RHO=ETA/SIGMA
      IF(RHO.LE.1.) GOTO 3
5     CONTINUE
      SIGMA=SIG(2)
      IF(SIGMA.EQ.0.) GOTO 10
      ETA=SIG(J)
      RHO=ETA/SIGMA
3     IF(RHO.LT..2) GOTO 4
      CEP(JJ)=.585*(1.+RHO)*SIG
      GOTO 10
4     CEP(JJ)=.6744*(1.+.24*RHO)*SIGMA
10    CONTINUE
      RETURN
      END
```

Figure 103. Subroutine CEPC Program Listing

Table XXXIV. List of Symbols for Subroutine CEPC

Mnemonic	Value	Units	Input	Output	Description
CEP(J)				X	Vector containing horizontal and vertical CEP
ETA					Used in computing ρ
$RH\bar{O}$					CEP is a function of ρ
SIG(I)					Vector containing the square root of the variances of x, y, and z at impact
SIGMA					Used in computing ρ

SECTION VI

CONCLUSIONS AND RECOMMENDATIONS

The overall objectives of this study were threefold: (1) development of theoretical analyses and mathematical models for precision weapon delivery, (2) development and documentation of computer analysis programs, and (3) demonstration of their use. The major emphasis has been on software development.

These objectives were primarily met. The analyses and model developments are reported in a separate document, Volume I. The developed programs have been carefully documented in Sections I through Section V.

Testing and demonstration of the use of the programs are reported in Volume III. Although an exhaustive parametric study could not be carried out due to lack of time, one example with a specified iron bomb and a representative tactical fighter-bomber aircraft was run to show the use of the programs.

In the following, the results and recommendations for future studies pertaining to the work reported in this volume are presented.

SIGNIFICANT RESULTS

- The work reported here established the total dynamic system approach to the analysis of weapon delivery problem.
- The chief benefit of the program is to provide software for rapid evaluation of system performance.
- Each subprogram (ADAP 1, 2 and 3) requires 32K of memory for a 17th-order system.

RECOMMENDATIONS FOR FUTURE SOFTWARE DEVELOPMENT WORK

Some of the interesting issues which arose in the course of the software development are listed below for future work:

- Improve the nonstationary performance evaluation program (ADAP 2) with respect to computing-time requirements. The computing cost can be reduced by more elaborate

programming (matrix partitioning), by using a different discretization technique, and by the Frobenius transformation. Exploit the special time-varying nature of data, $A(t) = A_0 + A_1 t$.

- To extend the performance evaluation capability, add cross covariance differential equations, as developed on page 167 of Volume I into the existing software for nonoptimal estimators.

- Improve CEP and SEP evaluations by integrating the probability density function of the states developed on page 133 of Volume I.

CONCLUSIONS

A large-scale system software for the analysis and design of precision weapon delivery systems is developed in this volume. The programs which implement the models developed in Volume I are documented with the user in mind.

REFERENCES

1. Bean, H. E., "General Purpose Aircraft Simulation System," Honeywell Aerospace Division, Minneapolis, AM-173, 22 April 1968.

2. Mueller, L., Bean H. E., "Digital Computer Handbook," Honeywell Aerospace Division, Minneapolis, AM-95, 11 November 1967.

APPENDIX I

TABLE INPUT, LOOK-UP AND INTERPOLATION PROCESSES IN ADAPS

The ADAP System contains a subroutine called FLOOK that is capable of inputting data tables of functions of one, two or three variables, as well as performing a table look-up and linear interpolation to compute function values from these tables [2].

The description of the subroutine FLOOK is briefly presented first in the following. The detailed description of the input, look-up and interpolation logics are given next. The flow charts, program listings, and symbol tables are presented on pages 125 through 139.

USAGE OF SUBROUTINE FLOOK

The number of functions the subroutine can handle is limited only by computer storage capacity. The subroutine will not extrapolate; i. e., it will not attempt to compute a function value beyond the range of its variables. The variables are effectively limited to the maximum and minimum values given in the data. In other words, if a function value is requested beyond the given range of its variable, the function value computed will be the function value at the last variable value in the given table. This constraint is shown graphically for a one-variable function in Figure I-1 with dotted lines.

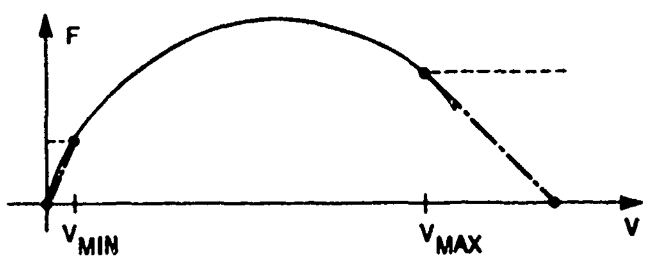


Figure I-1. One-Variable Function Constraint

If the data is not available beyond V_{max} in Figure I-1, then the user can extrapolate with a ruler as shown by the dotted line and add the extra point to the input. Otherwise the function will remain constant after the variable V passes the last stored value, V_{max}.

Execution time required to generate a table function is nearly independent of the number of points stored because of the nature of the functions that are normally being considered. These functions are continuous and normally the change in variable values between successive table look-ups are small enough that it doesn't bypass more than one stored point. (If the variables change faster than this, the functions are not being generated often enough.) Going on this assumption, the program saves the variable values from the last table look-up and starts from there.

The function look-up subroutine can be instructed to compute values for all functions at once, any continuous block of functions, or a single function. It is most expedient, timewise, to make as few calls to this subroutine as possible; in other words, compute as many functions as possible on each call. However, it is also time-effective to generate slow-varying functions at a slower rate than faster-varying functions. For this purpose, it pays to organize the functions in blocks according to their rate of change with respect to time.

To make use of this subroutine the user must:

- Adjust dimensions in the subroutine.
- Set up correct dimensions and calling sequence in one or more subprograms.
- Punch function data onto cards.

The required dimensions are denoted in subroutine FLOOK with comment cards. This part of the subroutine is shown in Figure I-2.

The dimensions on the arrays VST and FUN are controlled in the calling subroutines since they appear in the call argument list. This means that the size to which they are dimensioned in FLOOK is not important; however, they must be dimensioned. The dimension sizes must be identical to the variable values set in subroutine FLOOK. For example, the comment cards in Figure I-2 say that the array IFST must be dimensioned to the value of MNFV. Since MNFV = 150, IFST is dimensioned to 150 in the DIMENSION statement. In each subroutine calling FLOOK, VST and FUN must be dimensioned to MNUV and MNF, respectively, where the values for MNUV and MNF are set in FLOOK. If more than one subroutine calls FLOOK, then each of the subroutines must have the common statement, COMMON VST, FUN, and each subroutine must dimension VST and FUN to the correct values.

```
      SUBROUTINE FLOOK(VST,FUN,ISTRT,IEND)
      COMMON/ADAP/MODE,A(1000)
      EQUIVALENCE(MFN,MNF),(TABRD,A(997))
      DIMENSION IFST(150),ILST(150),IFN(80),F(2500),LID(150),V(500)
      DIMENSION VST(20),NL(150),DLT(3),NFID(80),IMS(100),KKV(3),KV(3)
      DIMENSION R(9),FUN(80),XF(3),RALF(20)
      INTEGER BLANK,RALF
      IF(MODE.EQ.-1.AND.TABRD.NE.0.) GOTO 5510
      IF(RNDM.EQ.123456.) GOTO 510
 5510 RNDM=123456.
C
C     THE ARRAYS SHOULD BE DIMENSIONED AS FOLLOWS
C     DIMENSION A(200),IFST(MNFV),ILST(MNFV),IFN(MNF),F(MNFVL),LID(MNFV),
C              V(MNVVL),VST(MNUV),NL(MNFV),DLT(3),NFID(MNF),IMS(MKKV),
C               KKV(3),KV(3),R(9),FUN(MNF),XF(3),RALF(20)
C     WHERE
   10 MKKV   = 100  (Max. No. of Variable Value Sets)
      MNFV   = 150  (Max. Total No. of Variables Specified)
      MNF=80        (Max. No. of Functions)
      MNFVL=2500    (Max. No. of Total Function Table Values)
      MNVVL=500     (Max. No. of Total Variable Values)
      MNUV   = 20   (Max. No. of Distinct Variables)
```

Figure I-2. Dimensioning of Subroutine FLOOK

Calling Sequence

All variables and functions are identified by numbers in the function data input. The numbers used for input are also used to identify the variables and functions in the calling subroutines. If a variable is assigned the interger i and a function the integer k for input purposes, they are identified in the calling subroutine by VST(i) and FUN(k), respectively.

Before FLOOK is called to generate function values, the required variable values must be transferred into the appropriate VST array position. After the return from FLOOK, the function values will be contained in the FUN array. As an example, suppose that functions identified by the integers 7, 8, 9, 10 and 14 are to be generated; the functions 7, 8, 9 and 10 are functions of variables identified by the integers 1, 3, 4, 7, 9 and 11; function 14 is a function of the two variables 5 and 6. The coding required to do this is:

$V(1) = V_1$

$V(3) = V_3$

$V(4) = V_4$

$V(7) = V_7$

$V(9) = V_9$

$V(11) = V_{11}$

CALL FLOOK (V, F, 7, 10)

$V(5) = V_5$

$V(6) = V_6$

CALL FLOOK (V, F, 14, 14)

where V_1, V_3, V_4, etc., represent the current values of variables. The function values will be contained in F (7), F (8), etc.

The general form of the CALL statement is

CALL FLOOK (V, F, ISTRT, IEND)

where

V - variable array

F - function array

ISTRT - number of first function to be generated

IEND - number of last function to be generated

How to Set Up Function Table Input

A function table is a table of function values tabulated over some matrix of variable values. The matrix will be either single-, double- or triple-dimensioned, depending on whether the function has 1, 2 or 3 variables. The variable values at which a function should be tabulated are left to the discretion of the user. The criteria which may be used to tabulate the aero data is to pick points on a curve so that a new curve constructed by drawing straight-line segments between the points which lie within ±10 percent of the original curve. This is fairly complicated for a three-variable function because it will be represented by families of curves. For example, F(M, α, δ) may be represented by the curves shown in Figure I-3.

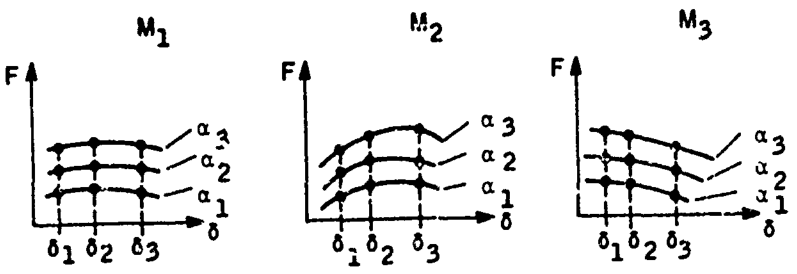


Figure I-3. Sample Function Curves

If the δ values chosen are δ_1, δ_2, and δ_3 then the matrix of variable values would be:

$(M_1, \alpha_1, \delta_1)$	$(M_1, \alpha_1, \delta_2)$	$(M_1, \alpha_1, \delta_3)$
$(M_1, \alpha_2, \delta_1)$	$(M_1, \alpha_2, \delta_2)$	$(M_1, \alpha_2, \delta_3)$
$(M_1, \alpha_3, \delta_1)$	$(M_1, \alpha_3, \delta_2)$	$(M_1, \alpha_3, \delta_3)$
$(M_2, \alpha_1, \delta_1)$	$(M_2, \alpha_1, \delta_2)$	$(M_2, \alpha_1, \delta_3)$
$(M_2, \alpha_2, \delta_1)$	$(M_2, \alpha_2, \delta_2)$	$(M_2, \alpha_2, \delta_3)$
$(M_2, \alpha_3, \delta_1)$	$(M_2, \alpha_3, \delta_2)$	$(M_2, \alpha_3, \delta_3)$
$(M_3, \alpha_1, \delta_1)$	$(M_3, \alpha_1, \delta_2)$	$(M_3, \alpha_1, \delta_3)$
$(M_3, \alpha_2, \delta_1)$	$(M_3, \alpha_2, \delta_2)$	$(M_3, \alpha_2, \delta_3)$
$(M_3, \alpha_3, \delta_1)$	$(M_3, \alpha_3, \delta_2)$	$(M_3, \alpha_3, \delta_3$

To identify the functions and variables on input cards they must be assigned numbers. It is also efficient to identify the variable value sets [for example (M_1, M_2, M_3)] by numbers because a single variable may assume different values for different function tables, and different variables may assume the same set of values. Therefore the first thing to do, once the function-tables are constructed, is to assign numbers to all function-tables, variables, and variable value sets.

For convenience -- and for storage capacity -- the numbering should start at 1 and proceed successively. The numbering in each of the three groups,

i.e., variables, variable value sets, and functions, should start with one. This means that a function, a variable, and a variable value set can all be assigned the same number and still be defined uniquely in the program. However, each variable, for example, must be assigned a unique number with respect to all other variables. As an example, the following assignments might be made for the sample function shown in Figure I-3:

$\alpha \rightarrow 1$

$F \rightarrow 3$

$M \rightarrow 7$

$\delta \rightarrow 3$

$(M_1, M_2, M_3) \rightarrow 3$

$(\alpha_1, \alpha_2, \alpha_3) \rightarrow 1$

$(\delta_1, \delta_2, \delta_3) \rightarrow 5$

To read the table input, a call to FLOOK is made in the initialization section (i.e., MODE = -1). The first time FLOOK is called, it reads input table data. The table data are placed after the RUN card in the input data deck.

The data for a function table is set up in one continuous block of cards which are made up of the four sub-blocks

1. Function header card
2. Variable-value cards
3. Function-table value cards
4. End function card

and must appear in that order.

Function Header Card -- This card identifies the table by the following data:

- Number of variables in the function
- Which variables are used in the function (their numbers)
- The variable value sets over which the function is tabulated
- Function number
- If one exists, the number of the previously specified function table which has exactly the same function values as the present function

The function header card format is shown in Table I-1, and a card for the example function is shown in Figure I-4.

Table I-1. Function Header Card Format

Columns	Entry
1-5	Number of variables in the function
6-9	Set of values for first variable
10-11	First variable
12-15	Set of values for second variable
16-17	Second variable
18-21	Set of values for third variable
22-23	Third variable
24-27	Function
28-31	Previous function with same values

NOTE: These are all integer fields and the entries must therefore be "right justified".

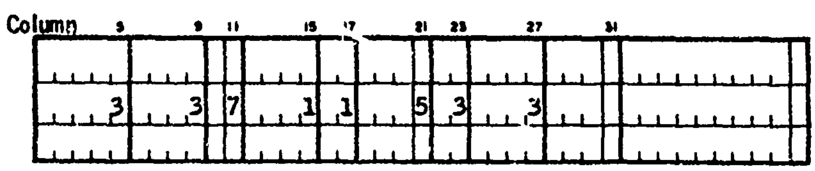


Figure I-4. Function Header Card Example

Variable Value Cards -- The numbers in each variable value set need to be specified only once. If a set is referenced on the function header card, it must either be given in this block or have been given in the variable-value section of a previously specified function-table. If a set is not referenced on the function header card, it cannot be specified in this section. If a particular set of values are specified in two different function-table blocks, the entry in the second block will be ignored.

Each card containing variable values must be identified by entering the set number on the card. Any number of cards may be used to specify a set of variable values. Each card has nine variable value fields as shown in Table I-2.

For the example function, suppose that set number 1 has been specified in a previous function table block, therefore, the cards for this variable-value section would be as shown in Figure I-5 for

$$(M_1, M_2, M_3) = (0.1, 0.3, 0.5)$$

$$(\delta_1, \delta_2, \delta_3) = (0.0, 10.0, 20.0)$$

Notice that the variable values shown are "left justified"; this is done to make it easier to check the data on the card, but is not necessary. The variable-value entries can be placed anywhere in a field, the only caution being that a decimal point must be specified somewhere in the field. The variable values must be in order of increasing value, i.e., with the least value first and the greatest value last.

Function Table Values -- Function table values are entered on exactly the same type of card as variable values except the field used for set number must always be blank. A function value must be given for each point of the matrix constructed from the variable-value sets. The function values must be specified in a certain order and that order is given in the sample matrix shown earlier, when reading from left to right and down. Therefore the function values must be in the order

$$F(M_1, \alpha_1, \delta_1)$$

$$F(M_1, \alpha_1, \delta_2)$$

$$F(M_1, \alpha_1, \delta_3)$$

$$F(M_1, \alpha_2, \delta_1)$$

$$F(M_1, \alpha_2, \delta_2) \quad \text{etc.}$$

In other words, the correct order is with the last variable varying fastest and the first varying slowest.

Table I-2. Variable-Value Card Format

Columns	Entry
1-4	Set number
9-16	Variable values
17-24	
25-32	
33-40	
41-48	
49-56	
57-64	
65-72	
73-80	

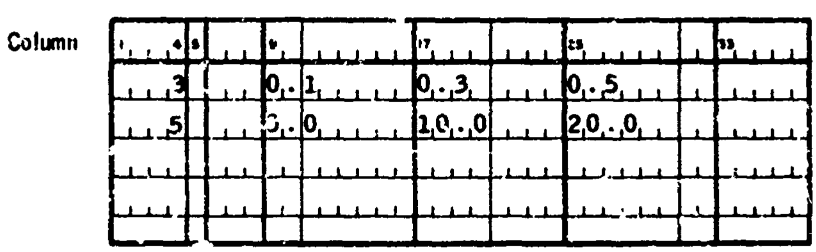


Figure I-5. Variable-Value Card Example

End Function Card -- This card has a -1 in columns 3 and 4 and signals the end of data for this function block. Any number of function blocks can be placed together and one extra -1 card must be placed after the last function table block.

Function-Table Input Card Set -- The complete set of cards prepared for function number 4, used in ADAP1, is shown in Figure I-8

TABLE DATA INPUT SECTION LOGIC

The function values for all function tables are stored in the single array called F in the order which they are read, and the variable values are stored in the single array V in the order which they are read. It is necessary to be able to locate the function values and variable values in these single arrays and therefore the beginning location of each function table, and the beginning and ending location of each variable value set are saved. The end location of a variable value set is saved because it is necessary to know the number of entires in each set. The beginning location for each function table is stored in the array IFN in the order which the functions appear in the input. If IFN(5) = 127, it means the fifth function read has its function table starting in F(127). The beginning and ending locations of the variable sets are stored in arrays IFST and ILST, respectively, in the order which the function variables occurred in the input. This information is not only entered in IFST and ILST when a variable-value set is read, but is also entered every time a variable references that set. Therefore, in the program the set number associated with a variable is not saved, but, instead, the beginning and ending locations of that set are saved.

To determine which entries in the IFST and ILST arrays go with each function, the number of variables in each function is stored in the NL array in the order which the functions are read. It is also necessary to know which variables are in each function so this information is also saved in the NL array by making the number of variables the hundreds digit in the entry and the variable numbers the 10 and 1's digit. Therefore, if a function has the three variables, 12, 3 and 21, then the NL array will have the entries 312, 303, and 321.

There is a one-to-one correspondence between the entries in the arrays IFST, ILST and NL, thus:

IFST(10) = 87

ILST(10) = 94

NL(10) = 214

contains the information that variable number 14 appears in a two-variable function and references the variable value set contained in V(87) through V(94)

NO. OF VARS KNV	1ST SET NO. KKV(1)	1ST VAR NO. KV(1)	2ND SET NO. KKV(2)	2ND VAR NO. KV(2)	3RD SET NO. KKV(3)	3RD VAR NO. KV(3)	FUNCT NO. KFN	SAME AS FUNCT NO. KFNS
3	7	2	8	3	9	1	4	

(a) FUNCTION HEADER CARD

SET NO. KNV1		VAR. VAL. 1 R(1)	VAR. VAL. 2 R(2)	VAR. VAL. 3 R(3)	VAR. VAL. 4 R(4)	VAR. VAL. 5 R(5)	VAR VAL. 6	VAR. VAL. 7	VAR. VAL. 8	VAR. VAL. 9
7		0.0	20000.0	50000.0						
8		6.0	12.0	16.0	20.0	24.0				
9		0.4	0.9	1.1	1.6	2.0				

(b) VARIABLE VALUE CARDS

Figure I-6. Function-Table Input Card Set

	FUNCTION VAL. 1 R(1)	FUNCTION VAL. 2 R(2)	FUNCTION VAL. 3 R(3)	FUNCTION VAL. 4 R(4)	FUNCTION VAL. 5 R(5)				
	.0069	.0057	.0058	.0036	.0024				
	.0067	.0052	.0054	.0035	.0023				
	.0064	.0046	.0052	.0033	.0023				
	.0058	.0030	.0[illegible]46	.0029	.0022				
	.0051	.0022	.0039	.0024	.0020				
	.0071	.0064	.0069	.0043	.0030				
	.0068	.0059	.0063	.0042	.0030				
	.0065	.0050	.0060	.0040	.0028				
	.0059	.0032	.0053	.0035	.0027				
	.0053	[illegible]	.0044	.0029	.0023				
	.0072	[illegible]	.0080	.0053	.0036				
	.0069	.0063	.0076	.0052	.0036				
	.0067	.0054	.0069	.0048	.0034				
	.0062	.0035	.0059	.0041	.0032				
	.0054	.0020	.0048	.0033	.0027				

(c) FUNCTION VALUE CARDS

1234								

(d) END FUNCTION CARD

Figure I-6. Function-Table Input Card Set (concluded)

for that particular function. The only information that is still lacking is which function the above information is associated with. To obtain this information the function numbers are stored in the NFID array in the order which they are read. To associate these function numbers with the IFST, ILST and NF arrays, it is necessary to start from the beginning, i.e., at IFST(1), ILST(1), NL(1) and NFID(1), and keep track of the number of variables in each function. For example, if a set of function tables are read in the order

$F(\alpha, \beta)$

$G(\gamma)$

$H(\delta, \epsilon, \alpha)$

where

$\alpha = (\alpha_{F1}, \alpha_{F2}, \alpha_{F3})$ in F

$\beta = (\beta_1, \beta_2, \beta_3, \beta_4)$

$\gamma = (\gamma_1, \gamma_2)$

$\delta = (\delta_1, \delta_2, \delta_3)$

$\epsilon = (\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4) = (\beta_1, \beta_2, \beta_3, \beta_4)$

$\alpha = (\alpha_{H1}, \alpha_{H2}, \alpha_{H3}, \alpha_{H4}, \alpha_{H5})$ in H

and the following numbers are assigned

$F \to 2$

$G \to 3$

$H \to 5$

$\alpha \to 2$

$\beta \to 3$

$\gamma \to 1$

$\delta \to 5$

$\epsilon \to 4$

$(\alpha_{F1}, \alpha_{F2}, \alpha_{F3}) \to 1$

$(\beta_1, \beta_2, \beta_3, \beta_4) \rightarrow 5$

$(\gamma_1, \gamma_2) \rightarrow 10$

$(\delta_1, \delta_2, \delta_3) \rightarrow 17$

$(\epsilon_1, \epsilon_2, \epsilon_3, \epsilon_4) \rightarrow 5$

$(\alpha_{H1}, \alpha_{H2}, \alpha_{H3}, \alpha_{H4}, \alpha_{H5}) \rightarrow 3$

then

NFID(1) = 2	IFST(1) = 1	ILST(1) = 3	NL(1) = 202
	IFST(2) = 4	ILST(2) = 7	NL(2) = 203
NFID(2) = 3	IFST(3) = 8	ILST(3) = 9	NL(3) = 101
	IFST(4) = 10	ILST(4) = 12	NL(4) = 305
NFID(3) = 5	IFST(5) = 4	ILST(5) = 7	NL(5) = 304
	IFST(6) = 13	ILST(6) = 17	NL(6) = 302

By starting at NFID(1), IFST(1), ILST(1) and NL(1), it is possible to determine that IFST(3), etc., corresponds to NFID(2), i.e., function number 3, by noting the 2 in the hundreds digit of NL(1) and NL(2) which says that IFST(1), IFST(2), ILST(1), ILST(2), NL(1) and NL(2) correspond to NFID(1).

The function values are stored in the F array, and the IFN array contains the starting location for each function table, therefore, for this example

IFN(1) = 1

IFN(2) = 13

IFN(3) = 15

One other array, called IMS, is used during data read to determine which variable value sets have already been read, and where they are stored. This array must be dimensioned at least as large as the largest set number used. Initially, before any data is read, this array is set to zero and then when a set is read, the IMS array location with index equal to the set number is equated to the index of the IFST array location which contains the starting location of the set in question. For example, IMS(14) = 5 means that variable-value set number 1′ has been read and is stored in the V array starting in V(k1) and ending in V(k2) where k1 = IFST(5) and k2 = ILST(5). For the sample input given above the IMS array would be

IMS(1) = 1

IMS(3) = 6

IMS(5) = 2

IMS(10) = 3

IMS(17) = 4

The Table Look-Up section of FLOOK makes use of an array called LID to expedite the process of locating variable values. This array contains information about the location, in the V array, of all variables during the previous table look-up. For example, the previous value of α in the function f_{20} may have had a value which laid between the two set values stored in V(24) and V(25), therefore, the LID location corresponding to α in f_{20} will contain a 24.

In other words, the location of the smallest of the two set values which bound the variable value is saved for all variables in all functions.

The correspondence between the variables and the LID array is the same as the correspondence between the variables and the IFST array. The LID array is initially set equal to the IFST array and this is done in the Read Data section.

The arrays and single variables shown in Tables I-3 and I-4 are used to store the data temporarily as it is read from cards and before it is stored permanently in the arrays discussed above.

Table I-3. Symbols Used to Read Function Header Card

Symbol	Usage
KNV	Number of variables
KKV(1) KKV(2) KKV(3)	Set numbers associated with the first, second and third variables, respectively
KV(1) KV(2) KV(3)	First, second and third variables, respectively
KFN	Function number
KFNS	Number of the previously read function table with identical function table data

Table I-4. Symbols Used to Read Variable- or Function-Value Cards

Symbol	Usage
KNV1	Set number
R(1) R(2) R(3) R(4) R(5) R(6) R(7) R(8) R(9)	Variable values or function values

If any of the Error Stops shown in the FLOOK flow chart prescribed in Section III are reached, then the following message is printed out:

```
FUNCTION TABLE DATA ERROR
FUNCTION NUMBER  n1
NUMBER OF VARIABLES = n2
V1     = n3
V2     = n4
V3     = n5
SET1   = n6
SET2   = n7
SET3   = n8
NVVL   = n9
NFV    = n10
NF     = n12
KNV1   = n13
```

Contained in this message are the values of all variables that could cause an Error Stop. The symbols n1, n2, ... etc., are used here to represent the numbers that will be printed out. The symbols in the message have the meanings given in Table I-5.

Table I-5. Symbols in Error Stop Message

Symbol	Meaning
V1 V2 V3	Variable numbers
SET1 SET2 SET3	Variable value set numbers
NVVL	Total number of variable values read, i.e., variable values in all sets
NFV	Number of function variables that have been read, i.e., the sum of KNVs for all function header cards (see Table I-3)
NFVL	Total number of function-table values that have been read for all functions
NF	Number of different functions that have been read
KNV1	Value of the set-number field on the last variable-value or function-value card (see Table I-4)

LOOK-UP AND INTERPOLATION SECTION LOGIC

This section of the subroutine makes use of the information stored during the reading of function tables to compute values for all functions contained in the input. The process used is to first locate the set values which bound the value of each variable in the function, find the neighboring function values which are stored in the tables, and then interpolate between these function values to obtain the approximate function value at the current value of the function variables.

The interpolation between function values is linear and is described in detail in the following paragraphs.

Interpolation Process

Let the function in question be denoted by $f(\alpha, \beta, \gamma)$ and suppose that function table values are given for the variable-value sets $(\alpha_1, \alpha_2, \alpha_3, \alpha_4)$, $(\beta_1, \beta_2, \beta_3)$ and $(\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5)$. Suppose also that at the current time

$\alpha_2 \le \alpha \le \alpha_3$, $\beta_1 \le \beta \le \beta_2$ and $\gamma_4 \le \gamma \le \gamma_5$. Graphically the situation would be something like Figure I-7, where γ_c is the current value of γ.

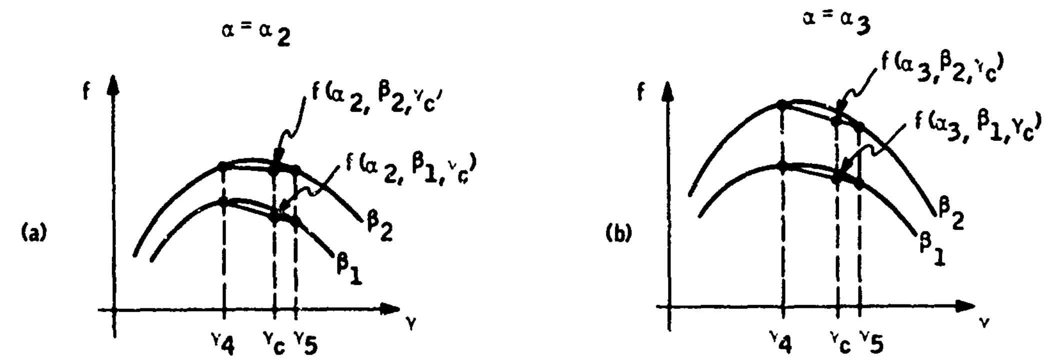


Figure I-7. Interpolation Process

Figure I-7 shows that a linear interpolation is used on the β_1 and β_2 curves to obtain values for $f(\alpha_2, \beta_1, \gamma_c)$, $f(\alpha_2, \beta_2, \gamma_c)$, $f(\alpha_3, \beta_1, \gamma_c)$ and $f(\alpha_3, \beta_2, \gamma_c)$. This can be written as

$$f(\alpha_2, \beta_1, \gamma_c) = f(\alpha_2, \beta_1, \gamma_4) + [f(\alpha_2, \beta_1, \gamma_5) - f(\alpha_2, \beta_1, \gamma_4)] \times \frac{\gamma_c - \gamma_4}{\gamma_5 - \gamma_4}$$

$$f(\alpha_2, \beta_2, \gamma_c) = f(\alpha_2, \beta_2, \gamma_4) + [f(\alpha_2, \beta_2, \gamma_5) - f(\alpha_2, \beta_2, \gamma_4)] \times \frac{\gamma_c - \gamma_4}{\gamma_5 - \gamma_4}$$

$$f(\alpha_3, \beta_1, \gamma_c) = f(\alpha_3, \beta_1, \gamma_4) + [f(\alpha_3, \beta_1, \gamma_5) - f(\alpha_3, \beta_1, \gamma_4)] \times \frac{\gamma_c - \gamma_4}{\gamma_5 - \gamma_4}$$

$$f(\alpha_3, \beta_2, \gamma_c) = f(\alpha_3, \beta_2, \gamma_4) + [f(\alpha_3, \beta_2, \gamma_5) - f(\alpha_3, \beta_2, \gamma_4)] \times \frac{\gamma_c - \gamma_4}{\gamma_5 - \gamma_4}$$

The next step is to use these four function values and interpolate for $f(\alpha_2, \beta_c, \gamma_c)$ and $f(\alpha_3, \beta_c, \gamma_c)$ where β_c is the current value of β. This interpolation can be written as

$$f(\alpha_2,\beta_c,\gamma_c) = f(\alpha_2,\beta_1,\gamma_c) + [f(\alpha_2,\beta_2,\gamma_c) - f(\alpha_2,\beta_1,\gamma_c)] \times \frac{\beta_c-\beta_1}{\beta_2-\beta_1}$$

$$f(\alpha_3,\beta_c,\gamma_c) = f(\alpha_3,\beta_1,\gamma_c) + [f(\alpha_3,\beta_2,\gamma_c) - f(\alpha_3,\beta_1,\gamma_c)] \times \frac{\beta_c-\beta_1}{\beta_2-\beta_1}$$

and finally

$$f(\alpha_c,\beta_c,\gamma_c) = f(\alpha_2,\beta_c,\gamma_c) + [f(\alpha_3,\beta_c,\gamma_c) - f(\alpha_2,\beta_c,\gamma_c)] \times \frac{\alpha_c-\alpha_2}{\alpha_3-\alpha_2}$$

By using the notation

$$\alpha_s = \frac{\alpha_c-\alpha_2}{\alpha_3-\alpha_2}, \quad \beta_s = \frac{\beta_c-\beta_1}{\beta_2-\beta_1} \text{ and } \gamma_s = \frac{\gamma_c-\gamma_4}{\gamma_5-\gamma_4},$$

and combining all of the above equations it is possible to write

$$\begin{aligned} f(\alpha_c,\beta_c,\gamma_c) = \{ & [(f(\alpha_3,\beta_1,\gamma_5) - f(\alpha_3,\beta_1,\gamma_4))\gamma_s + f(\alpha_3,\beta_1,\gamma_4) \\ & - (f(\alpha_2,\beta_1,\gamma_5) - f(\alpha_2,\beta_1,\gamma_4))\gamma_s - f(\alpha_2,\beta_1,\gamma_4)] (1-\beta_s) \\ & + [(f(\alpha_3,\beta_2,\gamma_5) - f(\alpha_3,\beta_2,\gamma_4))\gamma_s + f(\alpha_3,\beta_2,\gamma_4) \\ & - (f(\alpha_2,\beta_2,\gamma_5) - f(\alpha_2,\beta_2,\gamma_4))\gamma_s - f(\alpha_2,\beta_2,\gamma_4)] \beta_s \} \alpha_s \\ & + [(f(\alpha_2,\beta_1,\gamma_5) - f(\alpha_2,\beta_1,\gamma_4))\gamma_s + f(\alpha_2,\beta_1,\gamma_4)] (1-\beta_s) \\ & + [(f(\alpha_2,\beta_2,\gamma_5) - f(\alpha_2,\beta_2,\gamma_4))\gamma_s + f(\alpha_2,\beta_2,\gamma_4)] \beta_s \end{aligned}$$

Notice that if α is constant, i.e., f is a two-variable function, then the term $\{\ \}\alpha_s$ is zero and only the remaining terms need to be computed. This is the way the interpolation computations are handled in subroutine FLOOK.

Using the above equation to interpolate for a three-variable function requires that the eight function values

$f(\alpha_2, \beta_1, \gamma_4)$

$f(\alpha_2, \beta_1, \gamma_5)$

$f(\alpha_2, \beta_2, \gamma_4)$

$f(\alpha_2, \beta_2, \gamma_5)$

$f(\alpha_3, \beta_1, \gamma_4)$

$f(\alpha_3, \beta_1, \gamma_5)$

$f(\alpha_3, \beta_2, \gamma_4)$

$f(\alpha_3, \beta_2, \gamma_5)$

be located in the stored function tables via table look-up process, which is discussed in what follows.

Table Look-up Process

The general idea used in the table look-up is that if the starting location of a table of function values is known and the function values are stored in the correct order, i.e., with the last variable varying fastest and the first varying slowest, it is possible to compute the location of $f(\alpha_i, \beta_j, \gamma_k)$, where α_i, β_j, and γ_k are contained in the variable value sets used to construct the table. If the starting location of the function table is denoted by LS_f and the function table contains values for the sets $(\alpha_1, \alpha_2, \ldots, \alpha_L)$, $(\beta_1, \beta_2, \ldots, \beta_M)$ and $(\gamma_1, \gamma_2, \ldots, \gamma_N)$ then the location of $f(\alpha_i, \beta_j, \gamma_k)$, call it $Lf_{i,j,k}$, is

$$Lf_{i,j,k} = LS_f + N[j-1 + M(i-1)] + (k-1)$$

Using this equation, it is possible to obtain the following equations which can be used to compute the locations of all eight function values:

$$Lf_{i+1,j,k} = Lf_{i,j,k} + NM$$

$$Lf_{i,j+1,k} = Lf_{i,j,k} + N$$

$$Lf_{i+1,j+1,k} = Lf_{i,j+1,k} + NM$$

$$Lf_{i,\, j+1,\, k+1} = Lf_{i,\, j+1,\, k} + 1$$

$$Lf_{i+1,\, j+1,\, k+1} = Lf_{i+1,\, j+1,\, k} + 1$$

$$Lf_{i,\, j,\, k+1} = Lf_{i,\, j,\, k} + 1$$

$$Lf_{i+1,\, j,\, k+1} = Lf_{i+1,\, j,\, k} + 1$$

With the function values located, the linear interpolation can be performed to obtain $f(\alpha_c, \beta_c, \gamma_c)$ as described previously.

The information required to compute the above function value locations was stored during input read. The beginning location of each function table was stored in the IFN array and the beginning and ending location of each variable value set was stored in the IFST and ILST arrays, respectively. Therefore, the values of L, M and N can be computed from the ILST and IFST arrays, in particular

L = ILST(n) - IFST(n) + 1

M = ILST(n+1) - IFST(n+1) + 1

N = ILST(n+2) - IFST(n+2) + 1

In the above formula, n shows the inputting sequence of the variable value sets occurred during the input. For example:

IFST(10) = 87

ILST(10) = 94

means that the tenth variable value set read starts at the location V(87) and ends at the location V(94).

The values for i, j, and k can be found by searching the respective variable value sets for the set values which bound the current variable value. The set values are stored in the V array, and the set associated with α, for example, is stored in V(k1) through V(k2), where k1 = IFST(n) and k2 = ILST(n). If $V(i1) \le \alpha_c \le V(i2)$, where $k1 \le i1 \le i2 \le k2$ then $i = i1 - k1 + 1$, and i1 will be stored in LID(n) to be used as a starting point the next time. The IFN, IFST and ILST arrays contain data stored in the order which the input was read, and can be unscrambled by use of the NFID array. If i1 < k1 or i2 > k2 then the function value computed will be the function value at the last variable value in the table (Figure I-1).

The final values of all functions after interpolation are contained in the FUN array, and their location in that array is defined by the numbers assigned to them. For example, the value of $C_T(\alpha, \delta_{e_i})$, which is assigned the number 7, can be found in FUN(7). The current value of each variable is stored in the VST array in the same manner. In summary, if the required variable values are set in the VST array before executing the Table Look-Up and Interpolation section then the result will be a corresponding function value, stored in the FUN array, for all function tables in storage.

APPENDIX II

DIAK -- PROGRAM FOR OPTIMIZATION OF STATIONARY SYSTEMS

The stationary optimization program (DIAK) is used in ADAPS to compute the steady-state optimal controller and the estimator gains for a frozen-time-point linear-data set.

Table II-1 contains the subroutines in this program, as well as references to the analytical developments in Volume I. The first group of subroutines are the basic subroutines. The second group corresponds to data manipulation subroutines. The third group is the auxiliary set of subroutines, and they are the same as those described in the ADAP2(DISCOP) program in Section IV.

In the following the input/output description is given first. Then the main program and its subroutines are presented.

DIAK INPUT/OUTPUT

INPUT DESCRIPTION

Input for DIAK is in the form of cards and/or data stored on a permanent disc file.

Card Data Input

The first group of cards to be read is cards 1-4 which provide basic program data. Their formats are shown in Table II-2.

The next input occurs in SDATA subroutine when IDATA $\neq 0$. In this case the matrices FF, GG1, GG3 and H2 are input by subroutine INPT. Subsequent inputs occur in the subroutine DATAGEN. When IREADC $\neq 0$, H, D, Q matrices are input by calling subroutine INPT. If IREADE $\neq 0$ the matrices W1, W2 and HH2 are input in the same manner. The next input may occur in subroutine CGAINS. If INPC = 1, a constant is read in under the FORMAT (G10.4). If INPC = 2 the initial controller gain matrix is read in by subroutine INPT. The last input may occur in subroutine EGAINS when INPE = 1. In this case the initial value of the estimation error covariance is read in.

The complete card data input deck for DIAK is shown in Figure II-1.

Table II-1. DIAK Subroutine Summary

Subroutine	Description	Flow Chart (Figure No.)	Program Listing (Figure No.)	Volume I Reference
DIAK (Main Program)	Stationary optimization	II-2	II-3	pp. 161-169
CGAINS	Costate and controller gains	II-4	II-5	pp. 162-164
CAL	Lyapunov algorithm	II-6	II-7	pp. 161-162
EGAINS	Estimator gains, error covariance matrix and state covariance matrix	II-8	II-9	---
SDATA	System data input	II-10	II-11	---
DATAGEN	Data generator for controller and estimator	II-12	II-13	---
MP	Matrix print	---	---	---
INPT	Matrix input	---	---	---
TDINVR	Matrix inverse	---	---	---

Table II-2. Format for DIAK Data Input Cards 1-4

Card/Format	Column	Quantity	Description
1/(4I2)	1-2	ITAPC	ITAPC = 0 ⇒ Controller gains are not computed
			ITAPC ≠ 0 ⇒ Controller gains are computed
	3-4	ITAPE	ITAPE = 0 ⇒ Estimator gains are not computed
			ITAPE ≠ 0 ⇒ Estimator gains are computed
	5-6	ITAPPF	Logical number for permanent disc file
	8	IDATA	IDATA = 0 ⇒ No linear data input through cards
			IDATA ≠ 0 ⇒ Linear data input through cards
2/(4I2)	1-2	IMAX	Maximum number of inner-loop iterations
	3-4	ITER	Maximum number of outer-loop iterations
	5-6	INPC	INPC = 1 ⇒ Starting costate matrix IC is input
			INPC = 2 ⇒ Starting gain matrix K is input
			INPC = 3 ⇒ Starting costate matrix is from the previous run in the memory
	7-8	INPE	INPE = 1 ⇒ Input initial error covariance matrix
			INPE = 2 ⇒ Initial covariance matrix is from previous run in the memory
3/(5I2)	1-2	NX	Number of state variables
	3-4	NU	Number of controls
	5-6	NR	Number of responses
	7-8	NM	Number of measurements
	9-10	NW	Number of disturbances
4/(3I2)	1-2	NDPTS	Data point for the frozen time point linear data
	3-4	IREADC	IREADC = 0 ⇒ D, H, Q matrices for controller are input
			IREADC ≠ 0 ⇒ D, H, Q matrices are in the memory
	5-6	IREADE	IREADE = 0 ⇒ W_1 W_2 HH2 matrices for estimator are input
			IREADE ≠ 4 ⇒ W_1 W_2 HH2 matrices are in the memory

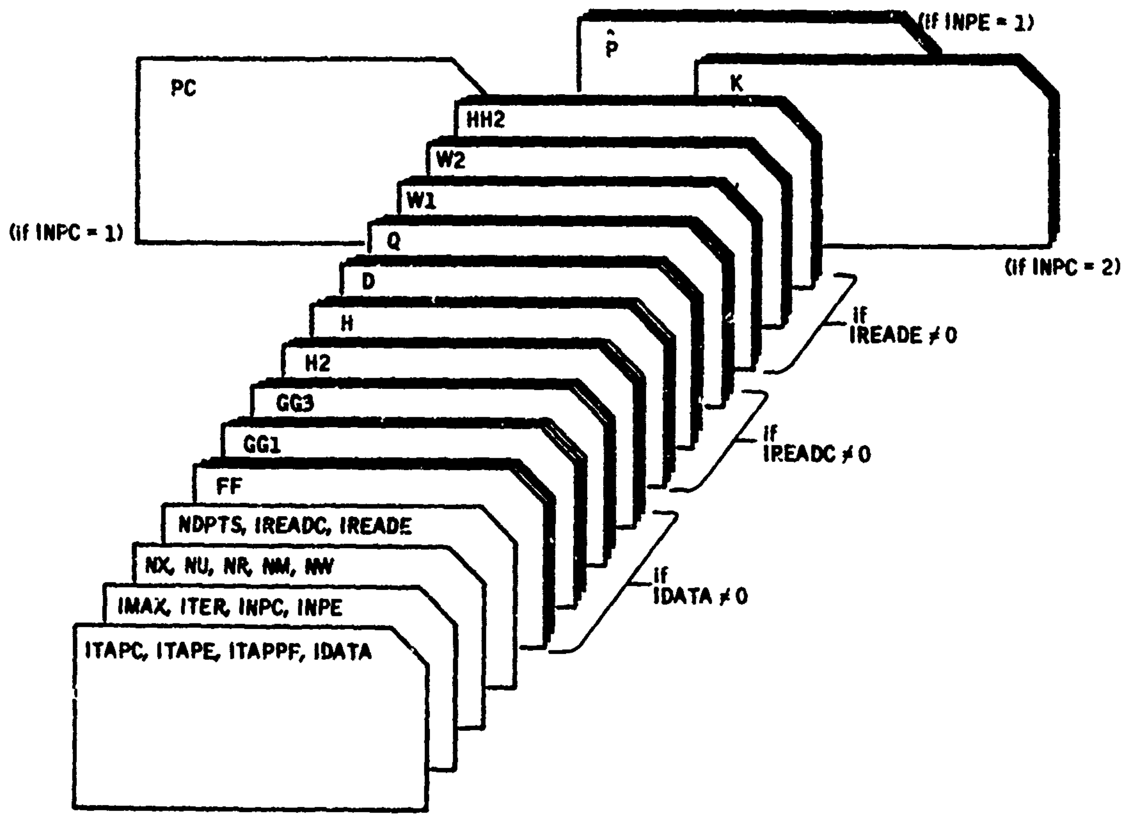


Figure II-1. DIAK Card Input Data Deck

Permanent Disc File Input

This type of input occurs first in subroutine DATAGEN. The linear data FF, GG1, GG2, GG3, H2 and VW are read in from the permanent disc file ITAPPF for the specified frozen-time point NDPTS. Subsequently the data written on the scratch tapes ITAPC and ITAPE are input in subroutine CGAINS and EGAINS, respectively.

OUTPUT DESCRIPTION

The output from DIAK is in the print form only. The parameters IMAX, ITER, NX, NR, NU, EE are printed out by subroutine CGAINS. Subsequently, the matrices F, G1, G2, H, D, A, E, Q are printed out. Then the costate matrix P and optimal gain matrix K are printed out. If convergence is not obtained, a message is printed out accordingly. The parameters corresponding to the estimator computations, IMAX, ITER, NX, NW, NM, EE are printed out by subroutine EGAINS.

Subsequently, the minimum error covariance matrix $\hat{P}$, optimal estimator gains L, the covariance of estimator $\hat{X}$, and the total covariance X are printed out using subroutine INPT.

PROGRAM DIAK DESCRIPTION

MAIN PROGRAM

Program DIAK generates for time-invariant systems the steady-state values of the optimal controller gains, optimal estimator gains as well as optimal error covariance and state covariance matrices. This program implements the analysis of Section X of Volume I.

The main program reads at the beginning the first four cards in the input data deck. If IDATA $\neq 0$ it also reads FF, GG1, GG3, and H_2 matrices by calling SDATA. Then for the given frozen-time point, the complete linear data for the controller and the estimator computations are prepared by subroutine DATAGEN. Calls to controller and estimator design subroutines are made depending on the ITAPC and ITAPE flags.

The design algorithms are double-iterative. If the solutions do not converge, exit occurs after a specified number of iterations.

After one cycle, the program goes back to the beginning and reads a new set of data. When it finds ITAPC = ITAPE = 0, it stops.

The flow diagram for program DIAK is shown in Figure II-2 and its program listing in Figure II-3.

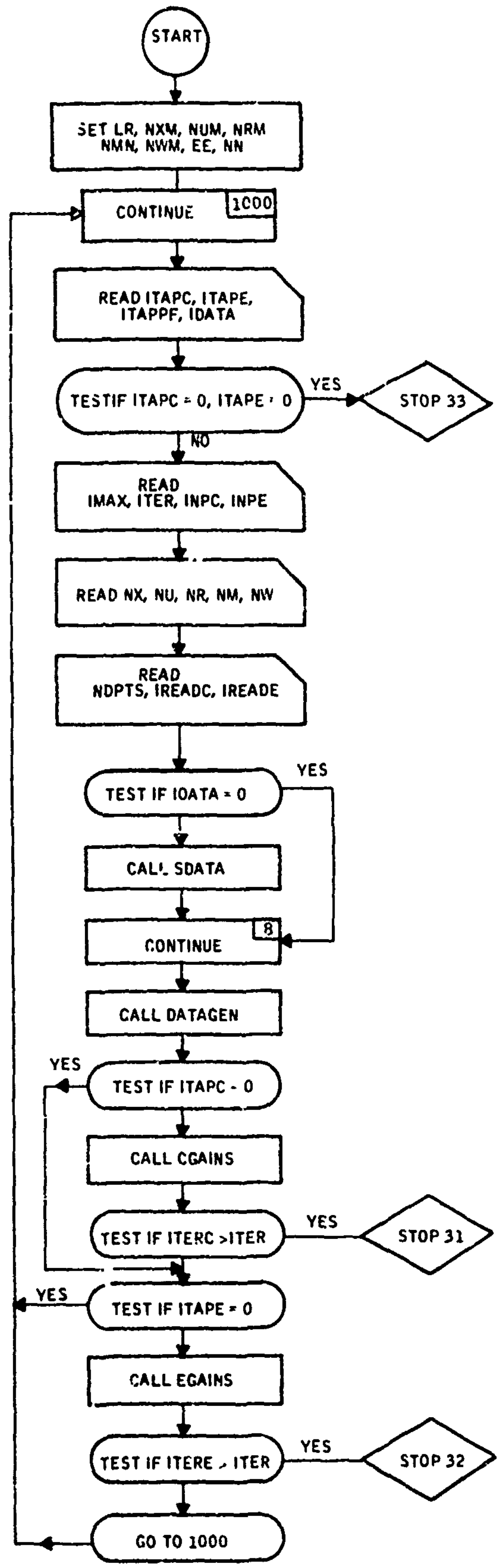


Figure II-2. DIAK Flow Diagram

```
      PROGRAM DIAK(INPUT,OUTPUT,TAPE5=INPUT,TAPE9=OUTPUT,TAPE6,TAPE7,TAP
     1F8)
C   MAIN PROGRAM DIAK
C
      COMMON NX,NU,NR,NW,NM,NXM,NUM,NRM,NWM,NMM,NN,EE
      DIMENSION AK(4,17)
      LR=5
      NXM=17
      NUM=4
      NRM=21
      NMM=12
      NWM=3
      EE=.001
      NN=3
 1000 CONTINUE
      READ(LR,1)ITAPC,ITAPE,ITAPPF,IDATA
    1 FORMAT(4I2)
      IF(ITAPC.EQ.0.AND.ITAPE.EQ.0) STOP 33
      READ(LR,2)IMAX,ITER,INPC,INPE
    2 FORMAT(4I2)
      READ(LR,3)NX,NU,NR,NM,NW
    3 FORMAT(5I2)
      REA[illegible])NDPTS,IREADC,[illegible]READE
    4 [illegible]OP[illegible]3I2)
      IF[illegible]A.EQ.0) GO [illegible] 8
      CALL [illegible]ATA(ITAPPF,NDPTS)
    8 CO[illegible]E
      CAL[illegible] (ITAPC,ITAPE,[illegible]APPF,NDPTS,IREADC,IREADE)
      IF([illegible] GOTO 10
      CALL C[illegible]S(AK,ITAPC,IMAX,ITER,ITERC,INPC)
      IF(ITER.LT.ITERC) STOP 31
   10 IF(ITAPE.EQ.0) GOTO 1000
      CALL EGAINS(AK,ITAPE,IMAX,ITER,ITERE,INPE)
      IF(ITER.LT.ITERE) STOP 32
      [illegible]TO 1000
      [illegible]
```

Figure II-3. DIAK Program Listing

BASIC SUBROUTINES

Subroutine CGAINS

Subroutine CGAINS generates the steady-state values of the costate and the controller gains in accordance with the analysis presented in Section X, of Volume I.

At the beginning of the program the equivalent matrices A, E and Q are generated, and all matrices involved in controller computation (F, G1, G2, H, D, A, E, Q) are printed out. There are three starting conditions in the iterative solution: (a) P_o = IC, (b) P_o is computed from K_o, and (c) P_o is set to what is already in the memory.

For convergence each distinct element of the symmetrical costate matrix P is subjected to the ratio test. If the NC elements pass the test, convergence is obtained, and normal exit occurs. The optimal gain matrix is computed and printed out along with the costate (Riccati) matrix. If convergence is not obtained, a message is printed out indicating the situation.

For each fixed right-hand side, the costate is computed by calling subroutine CAL.

The subroutine CGAINS flow diagram is shown in Figure II-4 and its program listing in Figure II-5.

Subroutine CAL

Subroutine CAL generates the Lyapunov matrix by solving iteratively the Lyapunov equation for IT = 1, and the covariance equation for IT = 2. The same test as described in subroutine CGAINS is used here for convergence. If convergence has not occurred in IMAX iterations, exit occurs with a message of iteration number. The subroutine CAL flow diagram is shown in Figure II-6 and its program listing in Figure II-7.

Subroutine EGAINS

Subroutine EGAINS generates the optimal estimator gains, minimum estimation error covariance and the optimal covariance of state as developed in Section X of Volume I.

The starting value $\hat{P}_o$ for the iterative solution is either entered (INPE = 1) or obtained from the previous solution left in the memory (INPE = 2). When convergence occurs, the error covariance matrix is printed out. Subsequently, the optimal gains and total covariance are completed and printed out.

The subroutine EGAINS flow diagram is shown in Figure II-8 and its program listing in Figure II-9.

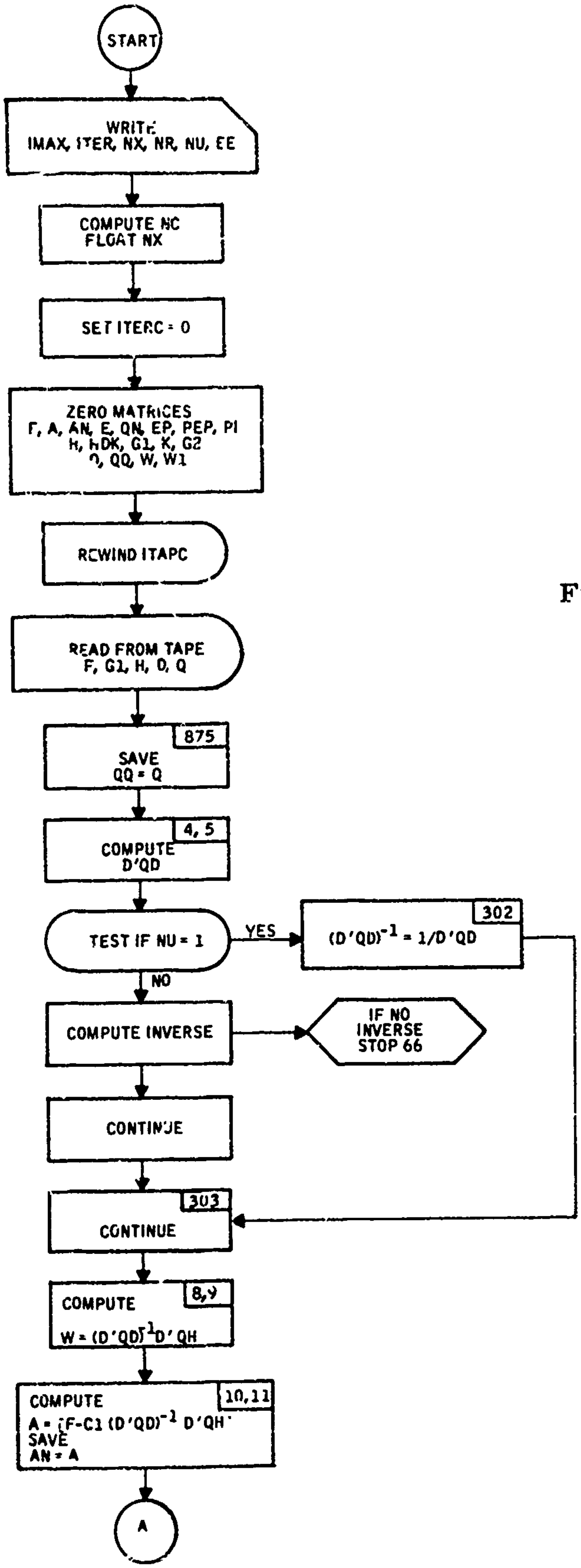


Figure II-4. Subroutine CGAINS Flow Diagram

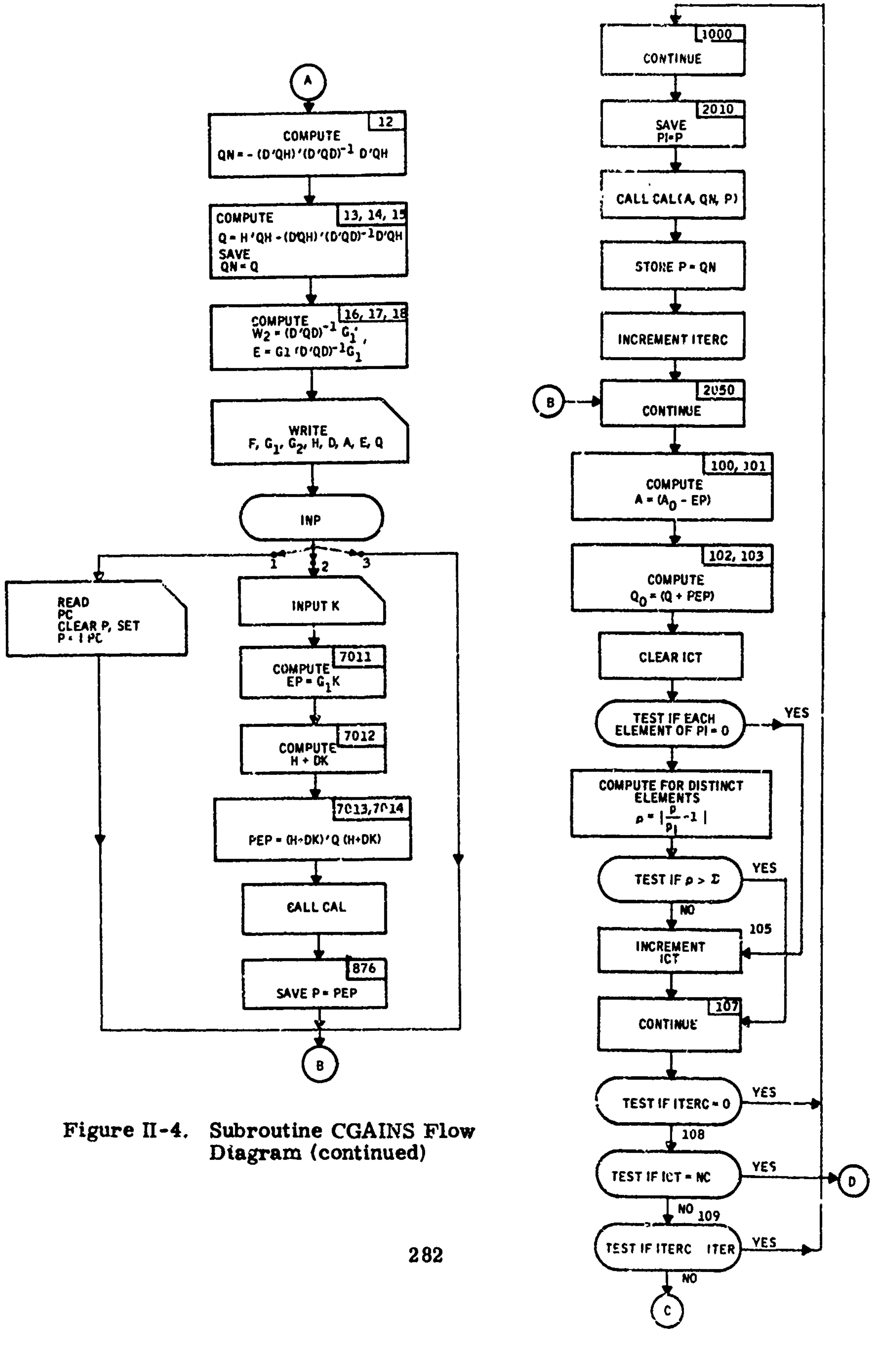


Figure II-4. Subroutine CGAINS Flow Diagram (continued)

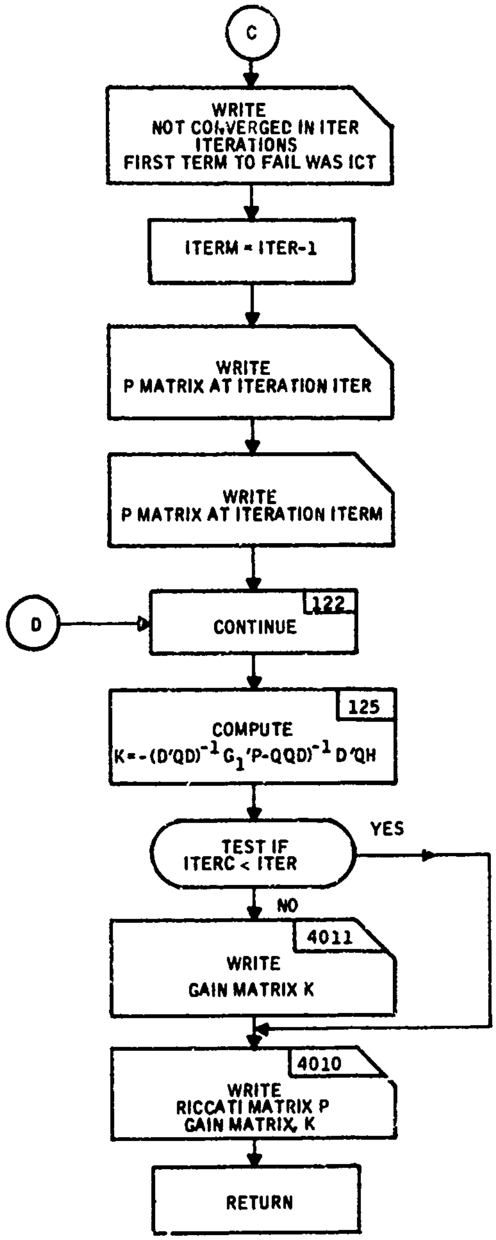


Figure II-4. Subroutine CGAINS Flow Diagram (concluded)

```
      SUBROUTINE CGAINS(AK,ITAPC,IMAX,ITER,ITERC,INP)
C DOUBLY-ITERATIVE ALGORITHM FOR SOLVING ALGEBRAIC RICCATI EQUATION
      COMMON NX,NU,NR,NW,NM,NXM,NUM,NRM,NWM,NMM,NN,EE
      DIMENSION F(17,17),G1(17,4),G2(17,3),AN(17,17),E(17,17)
      DIMENSION Q(21,21),QN(17,17),EP(17,17),PEP(17,17),P(17,17)
      DIMENSION H(21,17),D(21,4),AK(4,17),PI(17,17),DQD(4,4),KWA(4)
      DIMENSION W(21,21),W1(21,21),HDK(21,17),KKWA(17),QQ(21,21)
      DIMENSION W2(4,17)
      DIMENSION A(17,17)
      WRITE(9,4002)IMAX,ITER
 4002 FORMAT(1H1/7X,37H MAX NUMBER OF INNER-LOOP ITERATIONS I3,37H MAX N
     1UMBER OF OUTER-LOOP ITERATIONS I3//)
      WRITE(9,4003)NX,NR,NU,EE
 4003 FORMAT(//7X,18H ORDER OF SYSTEM =I3/7X,22H NUMBER OF RESPONSES =I3
     1/7X,21H NUMBER OF CONTROLS =I3/7X,18H CONVERG. FACTOR =F10.8//)
      NC=(NX*(NX+1))/2
      FN=NX
C  CALCULATE  A,E,Q IN RICCATI EQUATION PA+A*P-PEP+Q=0
C
C ZERO ARRAYS
C
 9099 CONTINUE
      ITERC=0
      DO 8020 I=1,NX
      DO 8013 J=1,NX
      F  (I,J)=0.
      A  (I,J)=0.
      AN (I,J)=0.
      F  (I,J)=0.
      QN (I,J)=0.
      EP (I,J)=0.
      PEP(I,J)=0.
      PI (I,J)=0.
 8013 CONTINUE
      DO 1700 J=1,NR
      H(J,I)=0.
      HDK(J,I)=0.
 1700 CONTINUE
      DO 8014 J=1,NU
      G1(I,J)=0.
 8014 AK(J,I)=0.
      DO 8015 J=1,NN
 8015 G2(I,J)=0.
 8020 CONTINUE
      DO 8045 I=1,NR
      DO 8045 J=1,NR
      Q(I,J)=0.
      QQ(I,J)=0.
      W(I,J)=0.
      W1(I,J)=0.
 8045 CONTINUE
      REWIND ITAPC
      READ(ITAPC)((F(I,J),J=1,NX),I=1,NX)
      READ(ITAPC)((G1(I,J),J=1,NU),I=1,NX)
```

Figure II-5. Subroutine CGAINS Program Listing

```
      READ(ITAPC)((H(I,J),J=1,NX),I=1,NR)
      READ(ITAPC)((D(I,J),J=1,NU),I=1,NR)
      READ(ITAPC)((Q(I,J),J=1,NR),I=1,NR)
      DO 875 I=1,NR
      DO 875 J=I,NR
      QQ(I,J)=Q(I,J)
      QQ(J,I)=Q(I,J)
  875 Q(J,I)=Q(I,J)
      DO 4 I=1,NU
      DO 4 J=1,NR
      W(I,J)=0.
      DO 4 K=1,NR
    4 W(I,J)=W(I,J)+D(K,I)*Q(K,J)
      DO 5 I=1,NU
      DO 5 J=1,NU
      DQD(I,J)=0.
      DO 5 K=1,NR
    5 DQD(I,J)=DQD(I,J)+W(I,K)*D(K,J)
      IF(NU-1)302,302,301
  302 DQD(1,1)=1./DQD(1,1)
      GOTO 303
  301 CONTINUE
      CALL TDINVR(ISOL,IDSOL,NU,NU,DQD,NU,KWA,DET)
      IF((ISOL+IDSOL)-2)6,6,7
    7 STOP 66
    6 CONTINUE
  303 CONTINUE
      DO 8 I=1,NU
      DO 8 J=1,NX
      W1(I,J)=0.
      DO 8 K=1,NR
    8 W1(I,J)=W1(I,J)+W(I,K)*H(K,J)
      DO 9 I=1,NU
      DO 9 J=1,NX
      W(I,J)=0.
      DO 9 K=1,NU
    9 W(I,J)=W(I,J)+DQD(I,K)*W1(K,J)
      DO 10 I=1,NX
      DO 10 J=1,NX
      A(I,J)=F(I,J)
      DO 11 K=1,NU
   11 A(I,J)=A(I,J)-G1(I,K)*W(K,J)
   10 AN(I,J)=A(I,J)
      DO 12 I=1,NX
      DO 12 J=1,NX
      QN(I,J)=0.
      DO 12 K=1,NU
   12 QN(I,J)=QN(I,J)-W1(K,I)*W(K,J)
      DO 13 I=1,NR
      DO 13 J=1,NX
      W1(I,J)=0.
      DO 13 K=1,NR
   13 W1(I,J)=W1(I,J)+Q(I,K)*H(K,J)
      DO 14 I=1,NX
```

Figure II-5. Subroutine CGAINS Program Listing (continued)

```
      DO 14 J=I,NX
      Q(I,J)=QN(I,J)
      DO 15 K=1,NR
   15 Q(I,J)=Q(I,J)+H(K,I)*W1(K,J)
      Q(J,I)=Q(I,J)
      QN(I,J)=Q(I,J)
   14 QN(J,I)=Q(I,J)
      DO 16 I=1,NU
      DO 16 J=1,NX
      W2(I,J)=0.
      DO 16 K=1,NU
   16 W2(I,J)=W2(I,J)+DQD(I,K)*G1(J,K)
      DO 17 I=1,NX
      DO 17 J=I,NX
      E(I,J)=0.
      DO 18 K=1,NU
   18 E(I,J)=E(I,J)+G1(I,K)*W2(K,J)
   17 E(J,I)=E(I,J)
      WRITE(9,20)
      CALL MP(NXM,NXM,NX,NX,F)
      WRITE(9,21)
      CALL MP(NXM,NUM,NX,NU,G1)
      WRITE(9,22)
      CALL MP(NXM,3,NX,3,G2)
      WRITE(9,23)
      CALL MP(NRM,NXM,NR,NX,H)
      WRITE(9,24)
      CALL MP(NRM,NUM,NR,NU,D)
      WRITE(9,25)
      CALL MP(NXM,NXM,NX,NX,A)
      WRITE(9,26)
      CALL MP(NXM,NXM,NX,NX,E)
      WRITE(9,27)
      CALL MP(NXM,NXM,NX,NX,QN)
   20 FORMAT(1H1/7X,10H F  MATRIX//)
   21 FORMAT(1H1/7X,10H G1 MATRIX//)
   22 FORMAT(1H1/7X,10H G2 MATRIX//)
   23 FORMAT(1H1/7X,10H H  MATRIX//)
   24 FORMAT(1H1/7X,10H D  MATRIX//)
   25 FORMAT(1H1/7X,10H A  MATRIX//)
   26 FORMAT(1H1/7X,10H E  MATRIX//)
   27 FORMAT(1H1/7X,10H Q  MATRIX//)
      GOTO(2000,3000,2050),INP
 2000 READ(5,2001)PC
 2001 FORMAT(G10.4)
      DO 2003 I=1,NX
      DO 2002 J=1,NX
 2002 P(I,J)=0.
 2003 P(I,I)=PC
      GO TO 2050
 3000 CALL INPT(AK,NUM,NXM)
      DO 7011 I=1,NX
      DO 7011 J=1,NX
      EP(I,J)=F(I,J)
```

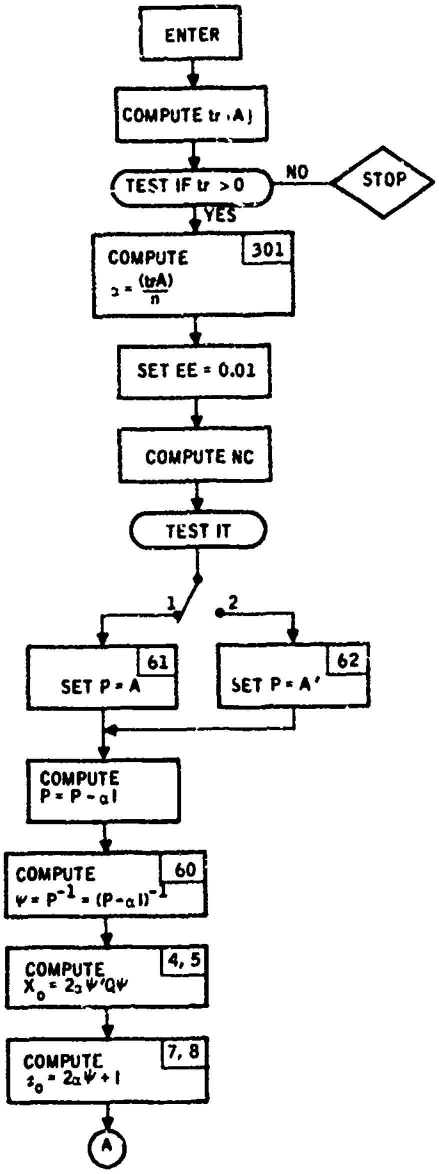
Figure II-5. Subroutine CGAINS Program Listing (continued)

```
      DO 7011 K=1,NU
 7011 EP(I,J)=EP(I,J)+G1(I,K)*AK(K,J)
      DO 7012 I=1,NR
      DO 7012 J=1,NX
      HDK(I,J)=H(I,J)
      DO 7012 K=1,NU
 7012 HDK(I,J)=HDK(I,J)+D(I,K)*AK(K,J)
      DO 7013 I=1,NR
      DO 7013 J=1,NX
      W1(I,J)=0.
      DO 7013 K=1,NR
 7013 W1(I,J)=W1(I,J)+QQ(I,K)*HDK(K,J)
      DO 7014 I=1,NX
      DO 7014 J=1,NX
      PEP(I,J)=0.
      DO 7014 K=1,NR
 7014 PEP(I,J)=PEP(I,J)+HDK(K,I)*W1(K,J)
      CALL CAL(EP,PEP,P,KKWA,NX,NXM,IMAX,1)
      DO 876 I=1,NX
      DO 876 J=1,NX
  876 P(I,J)=PEP(I,J)
      GO TO 2050
 1000 CONTINUE
      DO 2010 I=1,NX
      DO 2010 J=I,NX
 2010 PI(I,J)=P(I,J)
      CALL CAL(A,QN,P,KKWA,NX,NXM,IMAX,1)
      DO 877 I=1,NX
      DO 877 J=1,NX
  877 P(I,J)=QN(I,J)
      ITERC=ITERC+1
 2050 CONTINUE
      DO 100 I=1,NX
      DO 100 J=1,NX
      EP(I,J)=0.
      DO 101 K=1,NX
  101 EP(I,J)=EP(I,J)+E(I,K)*P(K,J)
  100 A(I,J)=AN(I,J)-EP(I,J)
      DO 102 I=1,NX
      DO 102 J=I,NX
      QN(I,J)=Q(I,J)
      DO 103 K=1,NX
  103 QN(I,J)=QN(I,J)+P(I,K)*EP(I,J)
  102 QN(J,I)=QN(I,J)
      ICT=0
      DO 105 I=1,NX
      DO 105 J=I,NX
      IF(PI(I,J))106,105,106
  106 RAT=P(I,J)/PI(I,J)-1.
      RAT=ABS(RAT)
      IF(RAT-EE)105,105,107
  105 ICT=ICT+1
  107 CONTINUE
      IF(ITERC)108,1000,108
```

Figure II-5. Subroutine CGAINS Program Listing (continued)

```
  108 IF(NC-ICT)109,122,109
  109 IF(ITERC-ITER)1000,1001,1001
 1001 WRITE(9,120)ITER,ICT
  120 FORMAT(1H1/7X,18H NOT CONVERGED IN I3,34H ITERATIONS-FIRST TERM TO
     1FAIL WAS I4/)
      ITERM=ITER-1
      WRITE(9,121)ITER
  121 FORMAT(///23H P MATRIX AT ITERATION I3//)
      CALL MP(NXM,NXM,NX,NX,P)
      WRITE(9,121)ITERM
      CALL MP(NXM,NXM,NX,NX,PI)
  122 CONTINUE
      DO 125 I=1,NU
      DO 125 J=1,NX
      AK(I,J)=-W(I,J)
      DO 125 K=1,NX
  125 AK(I,J)=AK(I,J)-W2(I,K)*P(K,J)
      IF(ITERC-ITER)4010,4011,4011
 4011 WRITE(9,4004)
 4004 FORMAT(1H1/7X,13H GAINS MATRIX//)
      CALL MP(NUM,NXM,NU,NX,AK)
 4010 WRITE(9,4005)
 4005 FORMAT(1H1/7X,15H RICCATI MATRIX//)
      CALL MP(NXM,NXM,NX,NX,P)
      WRITE(9,4004)
      CALL MP(NUM,NXM,NU,NX,AK)
      RETURN
      END
```

Figure II-5. Subroutine CGAINS Program Listing (concluded)



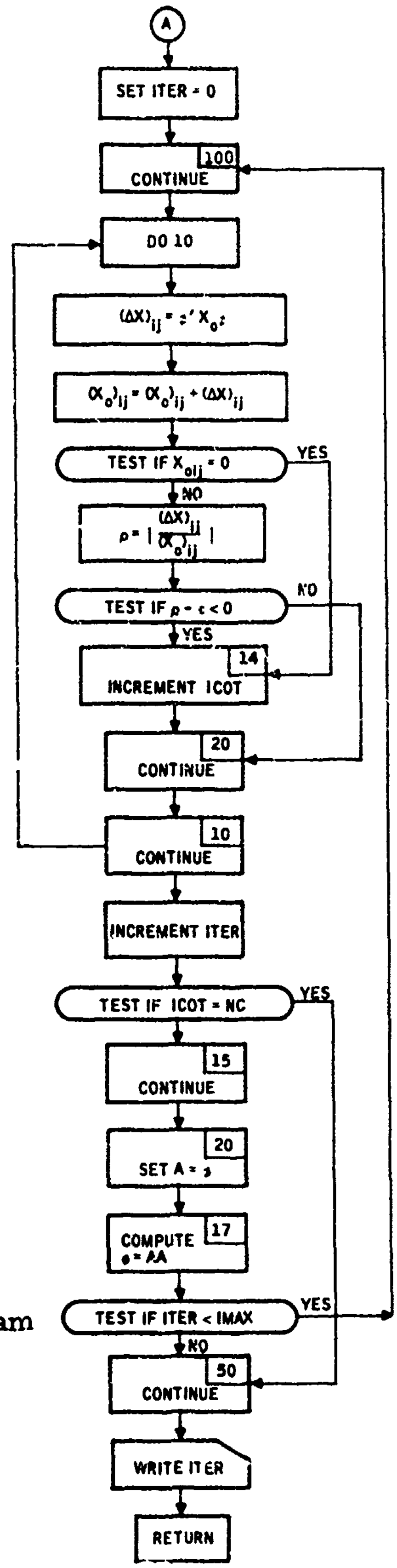


Figure II-6. Subroutine CAL Flow Diagram

```
      SUBROUTINE CAL(A,XN,P,KWA,N,NR,IMAX,IT)
      DIMENSION A(NR,1),XN(NR,1),P(NR,1),KWA(NR)
      TR=0.
      DO 300 I=1,N
  300 TR=TR+A(I,I)
      FN=N
      IF(TR)301,2,2
    2 STOP 66
  301 ALF=ABS(TR)/FN
      EE=.01
      NC=N*(N+1)
      NC=NC/2
      DO 60 I=1,N
      DO 63 J=1,N
      GOTO(61,62),IT
   61 P(I,J)=A(I,J)
      GOTO 63
   62 P(I,J)=A(J,I)
   63 CONTINUE
      P(I,I)=P(I,I)-ALF
   60 CONTINUE
      CALL TDINVR(ISOL,IDSOL,N,N,P,NR,KWA,DET)
      DO 4 I=1,N
      DO 4 J=1,N
      A(I,J)=0.
      DO 4 K=1,N
    4 A(I,J)=A(I,J)+P(K,I)*XN(K,J)*2.*ALF
      DO 5 I=1,N
      DO 5 J=1,N
      XN(I,J)=0.
      DO 5 K=1,N
    5 XN(I,J)=XN(I,J)+A(I,K)*P(K,J)
      DO 7 I=1,N
      DO 8 J=1,N
    8 P(I,J)=P(I,J)*2.*ALF
    7 P(I,I)=P(I,I)+1.
      ITER=0
  100 CONTINUE
      DO 9 I=1,N
      DO 9 J=1,N
      A(I,J)=0.
      DO 9 K=1,N
    9 A(I,J)=A(I,J)+P(K,I)*XN(K,J)
      ICOT=0
      DO 10 I=1,N
      DO 10 J=I,N
      DXIJ=0.
      DO 11 K=1,N
   11 DXIJ=DXIJ+A(I,K)*P(K,J)
      XN(I,J)=XN(I,J)+DXIJ
      XN(J,I)=XN(I,J)
      IF(XN(I,J))201,14,201
  201 RAT=ABS(DXIJ/XN(I,J))
      IF(RAT-EE)14,14,70
   14 ICOT=ICOT+1
   70 CONTINUE
   10 CONTINUE
   18 ITER=ITER+1
      IF(ICOT-NC)15,50,15
   15 CONTINUE
      DO 20 I=1,N
      DO 20 J=1,N
   20 A(I,J)=P(I,J)
   16 DO 17 I=1,N
      DO 17 J=1,N
      P(I,J)=0.
      DO 17 K=1,N
   17 P(I,J)=P(I,J)+A(I,K)*A(K,J)
   40 IF(ITER-IMAX)100,50,50
   50 CONTINUE
      WRITE(9,600) ITER
  600 FORMAT(/7X,6H ITER=I2)
      RETURN
      END
```

Figure II-7. Subroutine CAL Program Listing

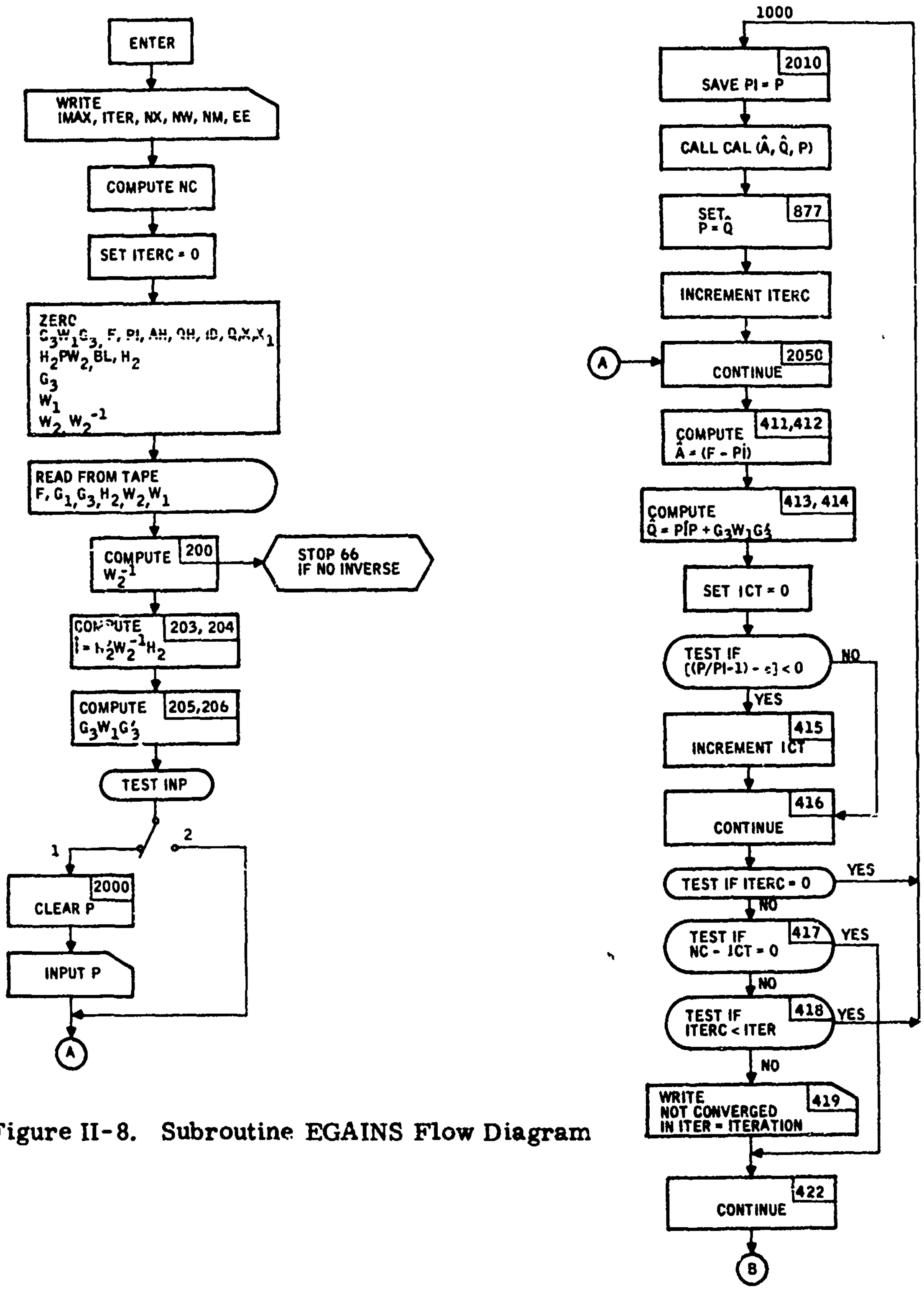


Figure II-8. Subroutine EGAINS Flow Diagram

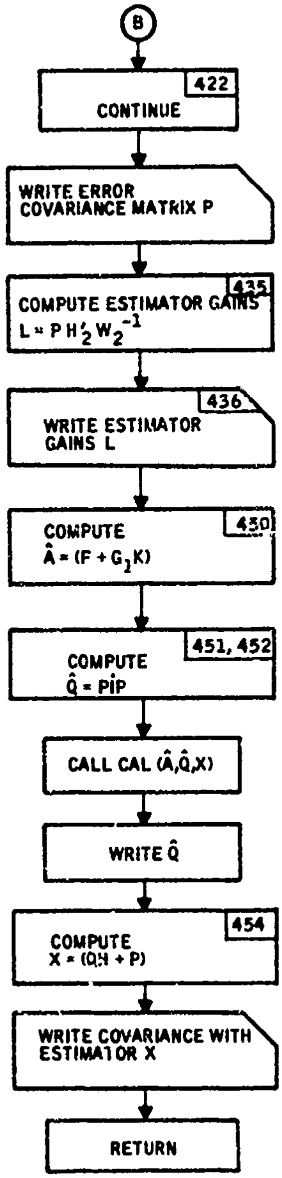


Figure II-8. Subroutine EGAINS Flow Diagram (concluded)

```
      SUBROUTINE EGAINS(AK,ITAPE,IMAX,ITER,ITERC,INP)
      COMMON NX,NU,NR,NW,NM,NXM,NUM,NRM,NWM,NMM,NN,EE
      DIMENSION F(17,17),G3(17,3),BL(17,12),H2(12,17),W2(12,12),W1(3,3)
      DIMENSION P(17,17),PI(17,17),AH(17,17),QH(17,17),ID(17,17)
      DIMENSION Q(17,17),X(17,17),X1(17,17),W2I(12,12),H2PW2I(17,12)
      DIMENSION G3W1G3(17,17),KWA(12),KKWA(17),AK(4,17),G1(17,4)
      REAL ID
      WRITE(9,4002)IMAX,ITER
 4002 FORMAT(1H1/7X,37H MAX NUMBER OF INNER-LOOP ITERATIONS I3,37H MAX N
     1UMBER OF OUTER-LOOP ITERATIONS I3//)
      WRITE(9,4003)NX,NW,NM,EE
 4003 FORMAT(//7X,18H ORDER OF SYSTEM =I3/7X,24H NUMBER OF DISTURBANCES=
     1I3/7X,25H NUMBER OF MEASUREMENTS =I3/7X,18H CONVERG. FACTOR =F10.8
     2//)
      NC=(NX*(NX+1))/2
 1099 CONTINUE
      ITERC=0
      DO 100 I=1,NX
      DO 101 J=1,NX
      G3W1G3(I,J)=0.
      F(I,J)=0.
      PI(I,J)=0.
      AH(I,J)=0.
      QH(I,J)=0.
      ID(I,J)=0.
      Q(I,J)=0.
      X(I,J)=0.
      X1(I,J)=0.
  101 CONTINUE
      DO 102 J=1,NM
      H2PW2I(I,J)=0.
      BL(I,J)=0.
      H2(J,I)=0.
  102 CONTINUE
      DO 103 J=1,NW
      G3(I,J)=0.
  103 CONTINUE
  100 CONTINUE
      DO 104 I=1,NW
      DO 104 J=1,NW
      W1(I,J)=0.
  104 CONTINUE
      DO 105 I=1,NM
      DO 105 J=1,NM
      W2(I,J)=0.
      W2I(I,J)=0.
  105 CONTINUE
      REWIND ITAPE
      READ(ITAPE)((F(I,J),J=1,NX),I=1,NX)
      READ(ITAPE)((G1(I,J),J=1,NU),I=1,NX)
      READ(ITAPE)((G3(I,J),J=1,NW),I=1,NX)
      READ(ITAPE)((H2(I,J),J=1,NX),I=1,NM)
      READ(ITAPE)((W2(I,J),J=1,NM),I=1,NM)
      READ(ITAPE)((W1(I,J),J=1,NW),I=1,NW)
```

Figure II-9. Subroutine EGAINS Program Listing

```
C
C COMPUTF W2 INVERSF,G3W1G3,H2PW2I,IDOT

      DO 200 I=1,NM
      DO 200 J=1,NM
  200 W2I(I,J)=W2(I,J)
      CALL TDINVR(ISOL,IDSOL,NM,NM,W2I,NMM,KWA,DET)
      IF((ISOL+IDSOL)-2)202,202,201
  201 STOP 66
  202 CONTINUE
      DO 203 I=1,NX
      DO 203 J=1,NM
      H2PW2I(I,J)=0.
      DO 203 K=1,NM
  203 H2PW2I(I,J)=H2PW2I(I,J)+H2(K,I)*W2I(K,J)
      DO 204 I=1,NX
      DO 204 J=1,NX
      ID(I,J)=0.
      DO 204 K=1,NM
  204 ID(I,J)=ID(I,J)+H2PW2I(I,K)*H2(K,J)
      DO 205 I=1,NX
      DO 205 J=1,NW
      X(I,J)=0.
      DO 205 K=1,NW
  205 X(I,J)=X(I,J)+G3(I,K)*W1(K,J)
      DO 206 I=1,NX
      DO 206 J=1,NX
      G3W1G3(I,J)=0.
      DO 206 K=1,NW
  206 G3W1G3(I,J)=G3W1G3(I,J)+X(I,K)*G3(J,K)
      GOTO(2000,2050),INP
 2000 CONTINUE
      DO 410 I=1,NX
      DO 410 J=1,NX
  410 P(I,J)=0.
      CALL INPT(P,NXM,NXM)
      GOTO 2050
 1000 DO 2010 I=1,NX
      DO 2010 J=1,NX
 2010 PI(I,J)=P(I,J)
      CALL CAL(AH,QH,P,KKWA,NX,NXM,IMAX,2)
      DO 877 I=1,NX
      DO 877 J=1,NX
  877 P(I,J)=QH(I,J)
      ITERC=ITERC+1
 2050 CONTINUE
      DO 411 I=1,NX
      DO 411 J=1,NX
      X(I,J)=0.
      DO 411 K=1,NX
  411 X(I,J)=X(I,J)+P(I,K)*ID(K,J)
      DO 412 I=1,NX
      DO 412 J=1,NX
  412 AH(I,J)=F(I,J)-X(I,J)
```

Figure II-9. Subroutine EGAINS Program Listing (continued)

```
      DO 413 I=1,NX
      DO 413 J=I,NX
      QH(I,J)=G3W1G3(I,J)
      DO 414 K=1,NX
  414 QH(I,J)=QH(I,J)+X(I,K)*P(K,J)
  413 QH(J,I)=QH(I,J)
      ICT=0
      DO 415 I=1,NX
      DO 415 J=I,NX
      IF(PI(I,J).EQ.0.) GOTO 415
      RAT=P(I,J)/PI(I,J)-1.
      RAT=ABS(RAT)
      IF(RAT-EE)415,415,416
  415 ICT=ICT+1
  416 CONTINUE
      IF(ITERC)417,1000,417
  417 IF(NC-ICT)418,422,418
  418 IF(ITERC-ITER)1000,419,419
  419 WRITE(9,430)ITER
  430 FORMAT(1H1/7X,18H NOT CONVERGED IN I3,11H ITERATIONS/)
  422 CONTINUE
      WRITE(9,431)
  431 FORMAT(1H1/7X,24H ERROR COVARIANCE MATRIX/)
      CALL MP(NXM,NXM,NX,NX,P)
      DO 435 I=1,NX
      DO 435 J=1,NM
      BL(I,J)=0.
      DO 435 K=1,NX
  435 BL(I,J)=BL(I,J)+P(I,K)*H2PW2I(K,J)
      WRITE(9,436)
  436 FORMAT(1H1/7X,16H ESTIMATOR GAINS/)
      CALL MP(NXM,NMM,NX,NM,BL)
      DO 450 I=1,NX
      DO 450 J=1,NX
      AH(I,J)=F(I,J)
      DO 450 K=1,NU
  450 AH(I,J)=AH(I,J)+G1(I,K)*AK(K,J)
      DO 451 I=1,NX
      DO 451 J=1,NX
      X(I,J)=0.
      DO 451 K=1,NX
  451 X(I,J)=X(I,J)+P(I,K)*ID(K,J)
      DO 452 I=1,NX
      DO 452 J=1,NX
      QH(I,J)=0.
      DO 452 K=1,NX
  452 QH(I,J)=QH(I,J)+X(I,K)*P(K,J)
      CALL CAL(AH,QH,X,KKWA,NX,NXM,IMAX,2)
      WRITE(9,453)
  453 FORMAT(1H1/7X,12H XHAT MATRIX/)
      CALL MP(NXM,NXM,NX,NX,QH)
      DO 454 I=1,NX
      DO 454 J=1,NX
  454 X(I,J)=QH(I,J)+P(I,J)
      WRITE(9,455)
  455 FORMAT(1H1/7X,34H COVARIANCE(WITH ESTIMATOR) MATRIX/)
      CALL MP(NXM,NXM,NX,NX,X)
      RETURN
      END
```

Figure II-9. Subroutine EGAINS Program Listing (concluded)

DATA MANIPULATION SUBROUTINES

Subroutine SDATA

Subroutine SDATA is called by the main program when the linear data is input through cards. It reads the linear shuffled data by calling INPT subroutine for FF, GG1, GG2, and H2 matrices. The data is stored in the permanent disc file ITAPPF for subsequent use by DATAGEN. The subroutine flow diagram is shown in Figure II-10 and its program listing in Figure II-11.

Subroutine DATAGEN

Subroutine DATAGEN generates linear data for the controller and estimator computations.

First, the linear data corresponding to the selected frozen-time point are read in from the permanent disc file. If IREADC = 0 the matrices D, H, Q are read in by calling subroutine INPT. Also if IREADE = 0 the matrices W1, W2, and HH2 are similarly read in.

The linear data for controller computations are FF, GG1, H, D, and Q. They are written on controller data scratch tape ITAPC. The linear data for estimator computations are FF, GG1. GG3, HH2, and W2, and W1. They are written on estimator data scratch tape ITAPE.

The flow chart of subroutine DATAGEN is shown in Figure II-12 and the program listing in Figure II-13.

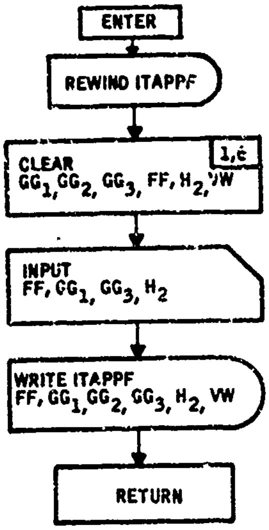


Figure II-10. Subroutine SDATA Flow Diagram

```
      SUBROUTINE SDATA(ITAPPF,NDPTS)
      COMMON NX,NU,NR,NW,NM,NXM,NUM,NRM,NWM,NMM,NN,EE
      DIMENSION FF(20,20),GG1(20,8),GG2(20,6),GG3(20,4),H2(21,12),VW(3)
      REWIND ITAPPF
    1 FF(I,J)=0.
      DO 1 I=1,20
      DO 2 J=1,8
    2 GG1(I,J)=0.
      DO 3 J=1,6
    3 GG2(I,J)=0.
      DO 4 J=1,4
    4 GG3(I,J)=0.
      DO 1 J=1,20
      DO 5 I=1,21
      DO 5 J=1,12
    5 H2(I,J)=0.
      DO 6 I=1,3
    6 VW(I)=0.
      CALL INPT(FF,20,20)
      CALL INPT(GG1,20,8)
      CALL INPT( GG3 ,20,4)
      CALL INPT(H2, 21,12)
      WRITE(ITAPPF)FF
      WRITE(ITAPPF)GG1
      WRITE(ITAPPF)GG2
      WRITE(ITAPPF)GG3
      WRITE(ITAPPF)H2
      WRITE(ITAPPF)VW
      RETURN
      END
```

Figure II-11. Subroutine SDATA Program Listing

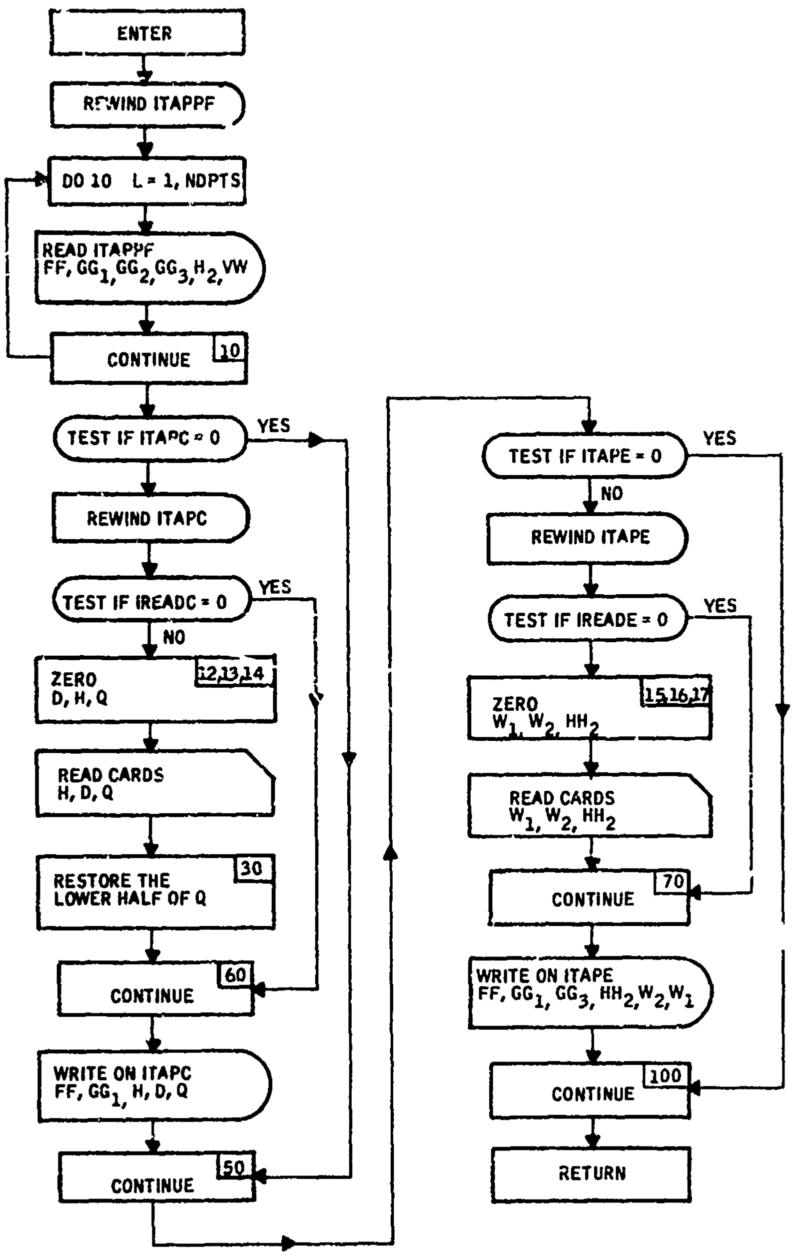


Figure II-12. Subroutine DATAGEN Flow Diagram

```
      SUBROUTINE DATAGEN(ITAPC,ITAPE,ITAPPF,NDPTS,IREADC,IREADE)
      COMMON NX,NU,NR,NW,NM,NXM,NUM,NRM,NWM,NMM,NN,EE
      DIMENSION FF(20,20),GG1(20,8),GG2(20,6),GG3(20,4),H2(21,12),VW(3)
      DIMENSION H(21,17),D(21,4),Q(21,21),W1(3,3),W2(12,12)
      DIMENSION HH2(12,17)
      REWIND ITAPPF
      DO 10 L=1,NDPTS
      READ(ITAPPF)FF
      READ(ITAPPF)GG1
      READ(ITAPPF)GG2
      READ(ITAPPF)GG3
      READ(ITAPPF)H2
      READ(ITAPPF)VW
   10 CONTINUE
C
C HERE WE CAN SHUFFLE DATA
C
      IF(ITAPC.EQ.0) GOTO 50
      REWIND ITAPC
      IF(IREADC.EQ.0) GOTO 60
      DO 11 I=1,NRM
      DO 12 J=1,NUM
   12 D(I,J)=0.
      DO 13 J=1,NXM
   13 H(I,J)=0.
      DO 11 J=1,NRM
   11 Q(I,J)=0.
      CALL INPT(H,NRM,NXM)
      CALL INPT(D,NRM,NUM)
      CALL INPT(Q,NRM,NRM)
      DO 30 I=1,NR
      DO 30 J=I,NR
   30 Q(J,I)=Q(I,J)
   60 CONTINUE
      WRITE(ITAPC)((FF(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPC)((GG1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPC)((H(I,J),J=1,NX),I=1,NR)
      WRITE(ITAPC)((D(I,J),J=1,NU),I=1,NR)
      WRITE(ITAPC)((Q(I,J),J=1,NR),I=1,NR)
   50 CONTINUE
      IF(ITAPE.EQ.0) GOTO 100
      REWIND ITAPE
      IF(IREADE.EQ.0) GOTO 70
      DO 15 I=1,NWM
      DO 15 J=1,NWM
   15 W1(I,J)=0.
      DO 16 I=1,NMM
      DO 16 J=1,NMM
   16 W2(I,J)=0.
      DO 17 I=1,NMM
      DO 17 J=1,NXM
   17 HH2(I,J)=0.
      CALL INPT(W1,NWM,NWM)
      CALL INPT(W2,NMM,NMM)
      CALL INPT(HH2,NMM,NXM)
   70 CONTINUE
      WRITE(ITAPE)((FF(I,J),J=1,NX),I=1,NX)
      WRITE(ITAPE)((GG1(I,J),J=1,NU),I=1,NX)
      WRITE(ITAPE)((GG3(I,J),J=1,NW),I=1,NX)
      WRITE(ITAPE)((HH2(I,J),J=1,NX),I=1,NM)
      WRITE(ITAPE)((W2(I,J),J=1,NM),I=1,NM)
      WRITE(ITAPE)((W1(I,J),J=1,NW),I=1,NW)
  100 CONTINUE
      RETURN
      END
```

Figure II-13. Subroutine DATAGEN Program Listing